Hindawi
Security and Communication Networks
Volume 2021, Article ID 5580272, 14 pages
https://doi.org/10.1155/2021/5580272

Research Article

Neural Reversible Steganography with Long Short-Term Memory

Ching-Chun Chang

Department of Computer Science, University of Warwick, Coventry CV4 7AL, UK

Correspondence should be addressed to Ching-Chun Chang; ching-chun.chang@warwickgrad.net

Received 19 February 2021; Revised 6 March 2021; Accepted 21 March 2021; Published 5 April 2021

Academic Editor: Chi-Hua Chen

Deep learning has brought about a phenomenal paradigm shift in digital steganography. However, there is as yet no consensus on the use of deep neural networks in reversible steganography, a class of steganographic methods that permits the distortion caused by message embedding to be removed. The underdevelopment of the field of reversible steganography with deep learning can be attributed to the perception that perfect reversal of steganographic distortion seems scarcely achievable, due to the lack of transparency and interpretability of neural networks. Rather than employing neural networks in the coding module of a reversible steganographic scheme, we instead apply them to an analytics module that exploits data redundancy to maximise steganographic capacity. State-of-the-art reversible steganographic schemes for digital images are based primarily on a histogram-shifting method in which the analytics module is often modelled as a pixel intensity predictor. In this paper, we propose to refine the prior estimation from a conventional linear predictor through a neural network model. The refinement can be to some extent viewed as a low-level vision task (e.g., noise reduction and super-resolution imaging). In this way, we explore a leading-edge neuroscience-inspired low-level vision model based on long short-term memory with a brief discussion of its biological plausibility. Experimental results demonstrated a significant boost contributed by the neural network model in terms of prediction accuracy and steganographic rate-distortion performance.

1. Introduction

Steganography is the art and science of concealing a message within a cover object (e.g., image, audio, video, and text) in an imperceptible manner [1]. Applications of modern steganography include copyright protection [2–4], tamper detection [5–7], covert communication [8–10], etc. The distortion caused by message embedding, albeit usually minimal and invisible, may to some extent contaminate the cover object. In this era of data-driven artificial intelligence, steganographic distortion might entail uncontrollable risks to the reliability of some autonomous machines since the robustness against steganographic distortion is probably not taken into consideration when building those machines. Accurate and consistent data lay a sound foundation of modern analytics platforms [11], and accordingly, the ability to reverse steganographic distortion and restore data integrity is of paramount importance.

Reversible steganographic methods have undergone rapid development over the past decades [12–22]. Although there are various principles and practices, a reversible steganographic method can be broadly compartmentalised into *coding* and *analytics* modules. In general, the coding module is devised to encode a message in an imperceptible and reversible way, whereas the analytics module exploits data redundancy with the aim of maximising steganographic capacity.

Deep learning has revolutionised both academia and industry [23]. The phenomenal advances in deep learning have also introduced a paradigm shift in digital steganography [24–29]. However, research on reversible steganography with deep neural networks remains largely undeveloped. A possible explanation might be that perfect reversal of steganographic distortion seems to be hardly achievable at first glance. A coding module often involves sophisticated designs and procedures in order to regulate imperceptibility and guarantee reversibility. Any faulty

operation may result in malfunctioning or failure of steganographic systems. A lack of transparency and interpretability in present neural networks could deter one from employing neural networks to realise or even upgrade these delicate reversible mechanisms. From our perspective, it is advisable to seek an alternative use of neural networks in reversible steganographic schemes. In contrast to the coding module, the analytics module has no demand for complete perfection, thereby allowing deep learning to serve its purpose. Recently, an exploratory study on adversarial learning for reversible image steganography was presented [30]. The author investigated a neural analytics module compatible with the regular singular (RS) coding module [31]. The neural analytics module was configured as a bit-plane predictor and implemented by a conditional generative adversarial network (GAN) called the *pix2pix* [32]. It has been suggested that transforming the analytics module into a neural network (*neuralisation*) could deliver a significant improvement to the original RS method.

Contemporary reversible steganographic schemes for digital images are based primarily on the histogram-shifting (HS) method on account of its sterling rate-distortion performance [33–40]. In general, this type of scheme consists of two procedures: histogram generation and histogram modification, linked to the analytics module and the coding module, respectively. The objective of histogram generation is to compute from an image a frequency distribution of which the data values are as concentrated as possible or, alternatively, the entropy is as small as possible. A more sharply distributed histogram would normally result in a finer steganographic rate-distortion performance. A simple example is the frequency distribution of pixel intensities. However, the distribution of pixel intensities is apparently diverse and not necessarily concentrated, and the entropy of such distribution might not be minimal. A better option is to consider the histogram of prediction errors. Providing a well-behaved predictor, the frequencies of prediction errors typically have a peak around zero and fall off exponentially from the peak on both sides (following a zero-mean Laplace distribution). The more accurate the predictor is, the more sharply distributed the histogram will become. To this end, scientists have proposed various approaches for pixel intensity prediction [41–46].

Given a fixed HS coding module, we can reasonably confine our attention to the design of an accurate pixel intensity predictor. Through experimental analysis, we found that although conventional (non-neural) predictors could estimate smooth image patches with a high degree of precision and are arguably less computationally demanding, their ability to predict textural patches is far from satisfactory. In view of this problem, we propose to employ a deep neural network model to refine *prior estimation* from a conventional predictor. While many deep neural network models may be employed to carry out the refinement, this task seems closest to low-level vision task (e.g., noise removal and super-resolution imaging) [47–53]. Therefore, we explore a seminal low-level vision model, the *MemNet* [54], of which the foundation is long short-term memory (LSTM) [55]. LSTM models were designed to mitigate the vanishing gradient problem encountered when training deep neural networks. The problem was overcome with the use of an internal mechanism called the *gate unit* which regulates the flow of information and learns to maintain important *hidden states* over extended time intervals. Although LSTM models are typically used for sequential data (e.g., time series, natural languages, and audio signals), the MemNet is a computer vision model that deals with low-level image features (e.g., edges, contours, and textures). Due to its state-of-the-art performance in image denoising and image super-resolution, we may reasonably expect to see an improvement delivered by the MemNet in the visual quality of pre-estimated images.

In this paper, we study a neural analytics module compatible with the HS coding module. While there are wide variations across HS methods (e.g., multiple histograms, multidimensional shifting, and optimal bin selection), we eliminate intricate mechanisms and focus on a prototype coding module in order to underline the performance gain contributed by the neural network model. The proposed neural analytics module comprises a pre-processing stage that generates a pre-estimated image via a linear predictor and a post-processing stage that refines the prior estimation via an LSTM-based vision model. Experimental results from large-scale assessments validated the effectiveness of the neural network model and demonstrated a significant improvement in steganographic rate-distortion performance.

The remainder of this paper is organised as follows. Section 2 reviews a prototype HS coding module and formulates some principal concepts. Section 3 presents the proposed neural analytics model which utilises an LSTM-based vision model for refining the prior estimation from a linear predictor. Section 4 validates the effectiveness of the neural network model and evaluates steganographic performance through simulation experiments. The paper draws conclusions in Section 5.

2. Coding Module

In this section, we revisit the coding module of a prototype HS method. We start with a workflow of the encoding and decoding processes, as illustrated in Figure 1. Suppose that a sender, Alice, wants to communicate a message to a receiver, Bob, through a reversible steganographic scheme. For a cover image X, Alice defines a set of *context* pixels preserved for predicting the other set of *query* pixels. The prediction can be fulfilled by either a conventional predictor or a neural network, resulting in a reference image $\widetilde{X}$. By subtracting $\widetilde{X}$ from X, cover residuals (prediction errors) are obtained. The HS coding module is applied to embed a message into cover residuals, yielding stego residuals along with an *overflow map* for later use in the reverse process. The stego image X' is finally generated by adding the stego residuals to $\widetilde{X}$. Addition may cause the problem of pixel intensity overflow; pixel intensities that are unexpectedly small or large wrap around the minimum and maximum after addition. In order to handle this exception, an overflow map is pre-calculated to flag pixels of which intensity would be off-boundary after

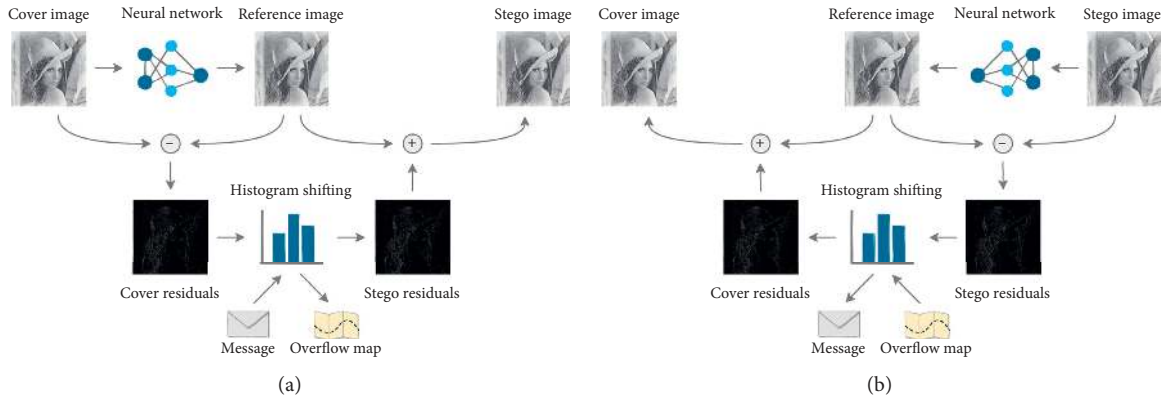


Figure 1: A workflow for a reversible steganographic scheme based on histogram shifting. (a) Encoding process. (b) Decoding process.

message embedding. The overall collection of sent data includes a stego image and a compressed overflow map. At the receiving end, Bob computes $\widetilde{X}$ from X' via a shared prediction mechanism. The reference image will be the same because only the query pixels have been modified and the context pixels in X and X' are unchanged. The remaining decoding procedures for message extraction and image recovery are virtually a reverse process of the encoding procedures. Next, we explain the details of the coding module under the assumption that the reference image $\widetilde{X}$ has already been obtained.

2.1. Histogram of Prediction Errors. Let us denote by $X_{i,j}$ a pixel at position (i, j) and $\widetilde{X}_{i,j}$ its predicted counterpart, where $X_{i,j}, \widetilde{X}_{i,j} \in [0, 255]$. For each query pixel, a prediction error is calculated by

$$E_{i,j} = X_{i,j} - \widetilde{X}_{i,j}, \tag{1}$$

where $E_{i,j} \in [-255, 255]$. Then, we count the occurrence of each error value and construct a histogram of prediction errors. We select one or more bins on the histogram as the *steganographic channel*. A bin is a container into which errors of the same value are grouped together. Selecting bins as the steganographic channel indicates defining which values of the prediction errors can be used to carry the message. In general, an increase in the number of selected bins will help to enhance steganographic capacity while simultaneously aggravating steganographic distortion. Let us denote by h_ε a bin for error value ε. According to the law of error [56], the frequency of an error could be expressed as an exponential function of its magnitude, disregarding its sign. In other words, small deviations would be observed more frequently than large deviations in normal circumstances. Hence, we may reasonably assume that the frequency of errors follows a zero-mean Laplacian distribution (i.e., double exponential distribution), in which the peak bin occurs around zero and the height of bins decays exponentially with the absolute magnitude of errors. Accordingly, we may explicitly define a channel selection rule that selects from h_0 and moves outwards in both positive and negative directions.

2.2. Encoding and Decoding. A summary for the HS coding mechanism is presented visually in Table 1. While the code chart allows us to develop a simpler understanding of the coding mechanism, we provide mathematical details to avoid confusion.

Let θ denote a threshold for the steganographic channel such that

$$\theta = \begin{cases} 0, & \text{if } h_0 \text{ is selected,} \\ 1, & \text{if } h_0 \text{ and } h_{\pm 1} \text{ are selected,} \\ 2, & \text{if } h_0,\ h_{\pm 1}, \text{ and } h_{\pm 2} \text{ are selected,} \\ \cdots & \end{cases} \tag{2}$$

According to the threshold, we derive the following three intervals:

$$\begin{aligned} \mathcal{U}_1 &= \{0\}, \\ \mathcal{U}_2 &= \{\pm 1,\ \pm 2,\ \pm 3, \ldots,\ \pm(\theta)\}, \\ \mathcal{U}_3 &= \{\pm(\theta+1),\ \pm(\theta+2), \ldots,\ \pm 255\}. \end{aligned} \tag{3}$$

The encoding process begins by shifting the bins selected as the steganographic channel (inner bins) and the remaining unselected bins (outer bins) outwards in order to empty out bins for carrying message digits. Shifting the inner and outer bins is equivalent to modifying prediction errors that fall into different intervals. We shift the value of each error by

$$E_{i,j}^{\Uparrow} = \begin{cases} 2E_{i,j}, & \text{if } E_{i,j} \in \mathcal{U}_1 \cup \mathcal{U}_2, \\ E_{i,j} + \operatorname{sgn}\left(E_{i,j}\right) \cdot (\theta + 1), & \text{if } E_{i,j} \in \mathcal{U}_3. \end{cases} \tag{4}$$

For an intended message, we divide it into two segments and convert them into the binary and ternary numeral systems, respectively. Then, we embed them depending on the error value that is currently observed. A pre-scanning is required in order to determine the length of each segment. Let us denote by m_{trit} a ternary message digit and by m_{bit} a binary message digit, where $m_{\text{trit}} \in \{-1, 0, 1\}$ and $m_{\text{bit}} \in \{0, 1\}$. We embed a ternary digit ($\log_2 3$ bits) if the error value is 0, embed a binary digit (1 bit) if the error value other than 0 originally falls into the steganographic channel, and skip the current error otherwise, as given by

Table 1: Code charts with different thresholds for the steganographic channel (grey cells).

							$\theta = 0$									
E	−254	⋯	−3		−2		−1	0		+1		+2		+3	⋯	+254
$E^{\Uparrow}$	−255	⋯	−4		−3		−2	0		+2		+3		+4	⋯	+255
E'	−255	⋯	−4		−3		−2	−1	0	+1	+2		+3		+4	⋯	+255
							$\theta = 1$									
E	−253	⋯	−3		−2		−1	0		+1		+2		+3	⋯	+253
$E^{\Uparrow}$	−255	⋯	−5		−4		−2	0		+2		+4		+5	⋯	+255
E'	−255	⋯	−5		−4	−3	−2	−1	0	+1	+2	+3	+4		+5	⋯	+255
							$\theta = 2$									
E	−252	⋯	−3		−2		−1	0		+1		+2		+3	⋯	+252
$E^{\Uparrow}$	−255	⋯	−6		−4		−2	0		+2		+4		+6	⋯	+255
E'	−255	⋯	−6	−5	−4	−3	−2	−1	0	+1	+2	+3	+4	+5	+6	⋯	+255

$$E'_{i,j} = \begin{cases} m_{\text{trit}}, & \text{if } E^{\Uparrow}_{i,j} \in \mathcal{U}_1, \\ E^{\Uparrow}_{i,j} + \text{sgn}\left(E^{\Uparrow}_{i,j}\right) \cdot m_{\text{bit}}, & \text{if } E^{\Uparrow}_{i,j} \in 2 \cdot \mathcal{U}_2, \\ E^{\Uparrow}_{i,j}, & \text{otherwise,} \end{cases} \tag{5}$$

where

$$2 \cdot \mathcal{U}_2 = \{\pm 2,\ \pm 4, \ldots,\ \pm 2\theta\}. \tag{6}$$

Finally, we add each modified prediction error to the estimated pixel at the corresponding position to obtain a stego image:

$$X'_{i,j} = \widetilde{X}_{i,j} + E'_{i,j}. \tag{7}$$

It is worth noting that pixel intensities after addition are not guaranteed within range of possible values from 0 to 255. Therefore, an overflow map is pre-calculated to flag pixels whose intensity might be out-of-bound. For pixels that may incur overflow, we skip the process of message embedding and record the positions by marking with flags on the map as

$$\Omega_{i,j} = \begin{cases} \perp (\text{false}), & \text{if } X'_{i,j} \in [0, 255], \\ \top (\text{true}), & \text{otherwise.} \end{cases} \tag{8}$$

The overflow map can be compressed and sent along or else embedded into the image as a part of the payload. For simplicity, we opt for the first approach in our implementation. Nevertheless, for fair evaluations, we deduct from the overall payload the size of the compressed overflow map when assessing steganographic capacity.

Decoding is simply the reverse process of encoding. It begins by generating the reference image $\widetilde{X}$ using the same set of context pixels as in the encoding process. For pixels where $\Omega_{i,j} = \perp$, we calculate the prediction errors by

$$E'_{i,j} = X'_{i,j} - \widetilde{X}_{i,j}. \tag{9}$$

Following the threshold and the coding mechanism, we divide pixels into the intervals:

$$\begin{aligned} \mathcal{U}'_1 &= \{0,\ \pm 1\}, \\ \mathcal{U}'_2 &= \{\pm 2,\ \pm 3, \ldots,\ \pm(2\theta + 1)\}, \\ \mathcal{U}'_3 &= \{\pm(2\theta + 2),\ \pm(2\theta + 3), \ldots,\ \pm 255\}. \end{aligned} \tag{10}$$

A ternary or binary digit is extracted based on different interval conditions such that

$$\begin{aligned} \widehat{m}_{\text{trit}} &= E'_{i,j} \quad \forall E'_{i,j} \in \mathcal{U}_1, \\ \widehat{m}_{\text{bit}} &\equiv E'_{i,j} \pmod 2 \quad \forall E'_{i,j} \in \mathcal{U}_2, \end{aligned} \tag{11}$$

and the cover image can be recovered by

$$\widehat{E}_{i,j} = \begin{cases} 0, & \text{if } E'_{i,j} \in \mathcal{U}'_1, \\ \left\lfloor \dfrac{E'_{i,j}}{2} \right\rfloor, & \text{if } E'_{i,j} \in \mathcal{U}'_2, \\ E'_{i,j} - \text{sgn}\left(E'_{i,j}\right) \cdot (\theta + 1), & \text{if } E'_{i,j} \in \mathcal{U}'_3. \end{cases} \tag{12}$$

3. Analytics Module

The previous coding module works under the assumption that a prediction mechanism has been developed and it is time to unveil and deliver the analytics module for estimating a reference image from the preserved context pixels. We begin by dividing pixels into the context and the query according to a pre-determined pattern. Next, we introduce a pre-processing stage for generating a *prior* reference image. Then, we explore a neural network model based on the long short-term memory for refining the pre-processed image into a *posterior* reference image.

3.1. Prior Estimation. The initial step of pixel prediction is typically to define the set of preserved pixels for estimating a query pixel, namely, the context. Amongst various ways to define the context and the query, the *chequerboard* pattern can be regarded as the most common one. Consider a chequerboard pattern that divides pixels into a black set and a white set, as illustrated in Figure 2. We may appoint the black set as the query and the white set as the context, or the other way round, which can be written mathematically as

$$X_{i,j} \in \begin{cases} \text{black (query)}, & \text{if } i + j \text{ is even}, \\ \text{white (context)}, & \text{if } i + j \text{ is odd}. \end{cases} \tag{13}$$

There are a variety of strategies for predicting the query pixels given the context pixels, but the most naïve strategy is

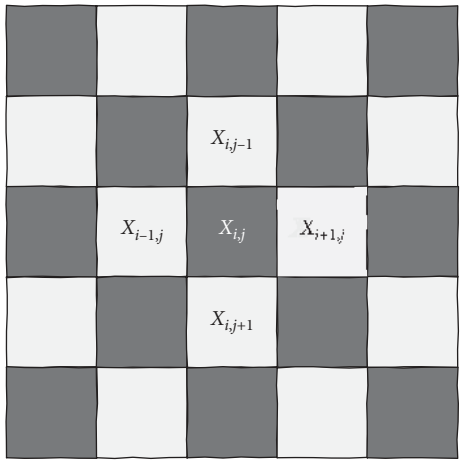


Figure 2: An illustration of the chequerboard pattern which divides pixels into the query and the context.

to estimate by the mean of four immediate context pixels, formulated as

$$\widetilde{X}_{i,j} = \frac{X_{i-1,j} + X_{i+1,j} + X_{i,j-1} + X_{i,j+1}}{4}. \tag{14}$$

This approach is, however, far from optimal due to a relatively restricted receptive field and limit of linearity. In other words, estimation is based solely on a linear combination of immediate local neighbours and any information outside the local field is completely ruled out.

In order to manage this issue, we may refine this pre-processed output by a nonlinear neural network model. We refer to the pre-processed image as the prior image $\widetilde{X}_{\text{pre}}$ and the refined image as the posterior image $\widetilde{X}_{\text{post}}$. Also, the pre-processor (linear non-neural model) is termed the prior predictor and the post-processor (nonlinear neural model) is termed the posterior predictor. We model this refinement process as a special type of low-level vision task and employ a vision model, the MemNet, to improve the visual quality of a pre-estimated reference image en route from input to output through hidden layers:

$$\widetilde{X}_{\text{post}} = \text{MemNet}\left(\widetilde{X}_{\text{pre}}\right). \tag{15}$$

Our implementation of the MemNet involves minor modifications. Consequently, the following description details the network architecture in order to ensure understanding, reproducibility, and replicability.

It is worth noting that the chequerboard-based prediction mechanism can be operated in two rounds, resulting in a dual-layer embedding scheme [57]. Suppose that the black set is assigned as the query and the white set as the context in the first round. After the first-layer embedding, the black set will be modified to carry a message segment. For the second-layer embedding, the white set will be assigned as the query and the modified black set as the context. Decoding is carried out in a first-in last-out manner; that is, pixels in the white set are recovered first and then used to recover pixels in the black set. We would like to emphasise that the dual-layer embedding scheme is not considered in our simulation experiments since our primary aim is to analyse the performance gain from neuralisation and an extended dual-layer embedding scheme would have few implications for the findings of this study.

3.2. Long Short-Term Memory. A fundamental component of the MemNet is the *memory cell*, which consists of neurons connected in a recurrent form and a gating mechanism that regulates persistent memories (i.e., important hidden states). From a practical and engineering standpoint, a slavish adherence to biological plausibility is not necessary for building neural network models; nonetheless, a neurobiological perspective may afford some interesting insights and provide guidance at a high level of abstraction [58]. Anatomical evidence has shown that recurrent synapses typically outnumber feedback and feedforward synapses, and it is believed that *recurrent circuitry* might play a crucial role in shaping the responses of neurons in the visual cortex [59]. Neuroscience studies also suggest that the mammalian brain has an evolved mechanism to avoid catastrophic forgetting called *synaptic consolidation*, whereby previously acquired knowledge, or memory, is durably encoded by rendering a proportion of synapses less plastic and thus stable over a long period of time [60].

Recurrent connections could be modelled as a recurrent neural network (RNN) [61]. For processing image data, it would be more convenient to construct a residual neural network (ResNet) [62] in such a way that the same weights are shared amongst layers. In fact, there is an intriguing equivalence between an RNN and a ResNet with weight sharing [63]. It can be seen from Figure 3 that a ResNet with weight sharing approximates an RNN when being unfolded into a feedforward network. Apart from a biological interpretation, recurrent connections can reduce the number of trainable parameters (i.e., weights and biases) substantially and thereby result in a comparatively *lightweight* model for storage. A gating mechanism mimicking synaptic consolidation could be represented by a convolutional layer that learns weights for preserving or erasing memories. After passing through the convolutional gate unit, a series of ephemeral recollections (short-term memories) become a recollection that persists (long-term memory).

Architectural details of the MemNet are described as follows. The MemNet is composed of a pre-processing layer f_{pre}, an LSTM module, and a post-processing layer f_{post}, as illustrated in Figure 4(a) and expressed symbolically by

$$\text{MemNet}(A) = f_{\text{post}}\left(\text{LSTM}\left(f_{\text{pre}}(A)\right) + A\right), \tag{16}$$

where f_{pre} and f_{post} are both convolutional layers with kernel size 3, stride 1, and padding 1. The post-processing layer takes not only the output of the LSTM module but also the original input. Shortcuts or skip connections are essential to deep neural networks. It has been shown that when the model gets deeper, skip connections allow the information from shallow layers to propagate more effectively to deep layers [64]. From our viewpoint, bypassing the intermediate layers and connecting the prior image directly to the last layer could guide the neural network to learn delicate textural information in images, namely, minute differences between the prior estimation and the *ground truth* (i.e., the pristine image). The distance between

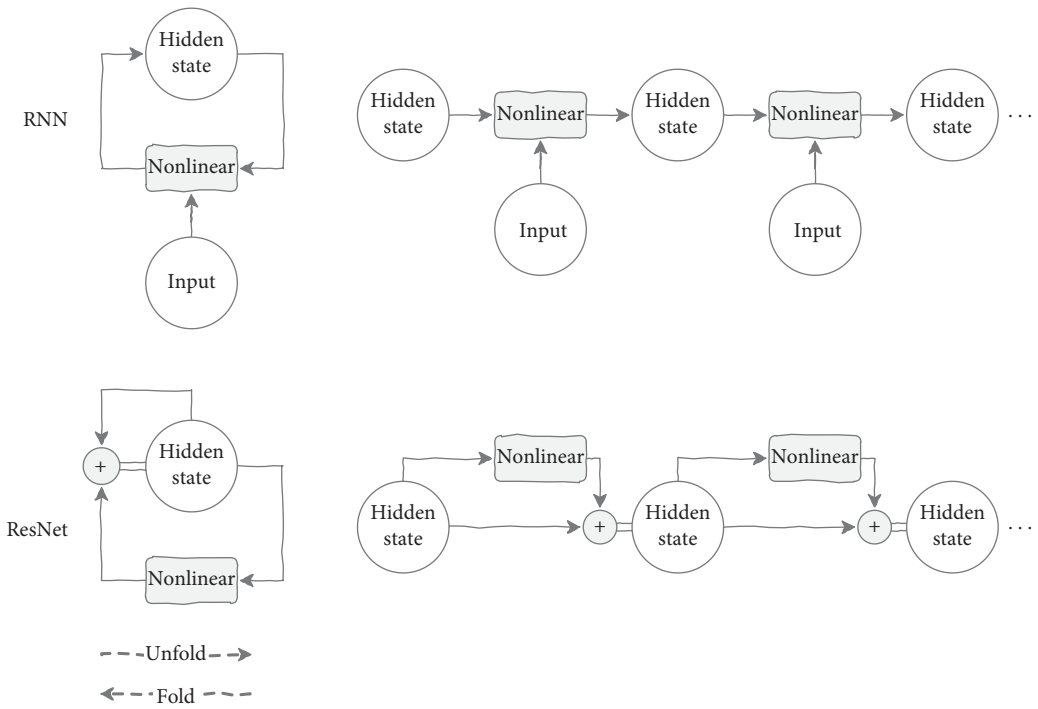


Figure 3: An equivalence relation between an RNN and a ResNet with weight sharing.

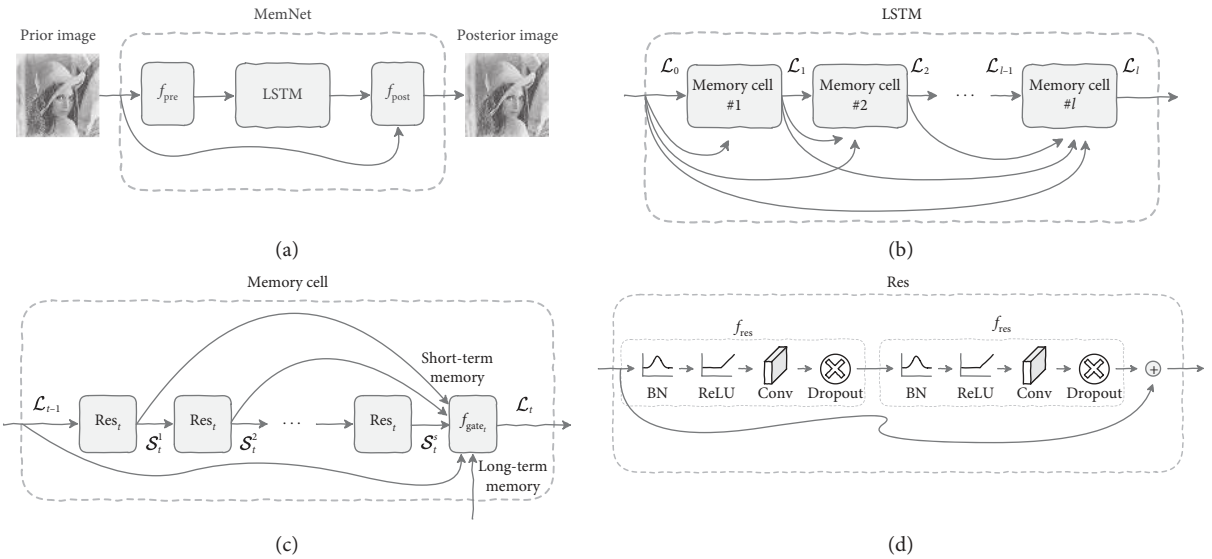


Figure 4: Architectural details of the MemNet. (a) MemNet. (b) Long short-term memory. (c) Memory cell. (d) Residual unit.

the refined output and the ground truth is measured by the ℓ_1 norm. The model is trained to minimise this loss function with the backpropagation algorithm [65].

The LSTM module comprises interconnected memory cells. Each current cell takes long-term memories produced from all previous cells as the input, as illustrated in Figure 4(b). Let l denote the number of memory cells and $\mathscr{L}_t$ the output from the t-th memory cell. The LSTM module inputs the 0-th memory and outputs the l-th memory:

$$\mathscr{L}_l = \mathrm{LSTM}(\mathscr{L}_0), \tag{17}$$

where

$$\mathscr{L}_0 = f_{\mathrm{pre}}(A). \tag{18}$$

A memory cell has several residual units connected to each other in a recurrent manner (with weight sharing) and a gate unit placed at the end of the cell, as illustrated in

Figure 4(c). The outputs from all residual units (i.e., short-term memories) along with the outputs from previous cells (i.e., long-term memories) go through a gate unit to produce a persistent memory for subsequent cells, as expressed by

$$\begin{aligned}\mathscr{L}_1 &= f_{\text{gate}_1}\left(\mathcal{S}_1^1 \| \mathcal{S}_1^2 \| \ldots \| \mathcal{S}_1^s \| \mathscr{L}_0\right),\\ \mathscr{L}_2 &= f_{\text{gate}_2}\left(\mathcal{S}_2^1 \| \mathcal{S}_2^2 \| \ldots \| \mathcal{S}_2^s \| \mathscr{L}_0 \| \mathscr{L}_1\right),\\ &\vdots\\ \mathscr{L}_l &= f_{\text{gate}_l}\left(\mathcal{S}_l^1 \| \mathcal{S}_l^2 \| \ldots \| \mathcal{S}_l^s \| \mathscr{L}_0 \| \mathscr{L}_1 \| \ldots \| \mathscr{L}_{l-1}\right),\end{aligned} \tag{19}$$

where

$$\begin{aligned}\mathcal{S}_t^1 &= \text{Res}_t\left(\mathscr{L}_{t-1}\right),\\ \mathcal{S}_t^2 &= \text{Res}_t\left(\text{Res}_t\left(\mathscr{L}_{t-1}\right)\right),\\ &\vdots\\ \mathcal{S}_t^s &= \text{Res}_t\left(\text{Res}_t\left(\ldots \text{Res}_t\left(\mathscr{L}_{t-1}\right)\ldots\right)\right).\end{aligned} \tag{20}$$

Residual unit is illustrated in Figure 4(d) and laid out as follows:

$$\text{Res}\left(A\right) = f_{\text{res}}\left(f_{\text{res}}\left(A\right)\right) + A. \tag{21}$$

The structure of both f_{res} and f_{gate} follows the basic building block, composed of a convolutional layer [66–68], a batch normalisation [69], a ReLU activation function [70], and a dropout regularisation [71], written as

$$f\left(A\right) = \text{Dropout}\left(\text{ReLU}\left(\text{BN}\left(\text{Conv}\left(A\right)\right)\right)\right). \tag{22}$$

In implementation, the convolutional layer of f_{res} was configured to kernel size 3, stride 1, and padding 1, whereas the convolutional layer of f_{gate} was set to kernel size 1, stride 1, and padding 1. We applied a dropout rate of 0.1 to f_{res} and f_{gate}.

4. Experimental Results

In this section, we present experimental results based on large-scale statistical evaluations. Our primary aim is to demonstrate the performance difference between the prior (linear non-neural) and posterior (nonlinear neural) predictors. We begin by validating the effectiveness of the neural network model for refining the visual quality of pre-estimated images. Then, we examine the error distribution with regard to entropy and cumulative frequency. In order to understand how the visual quality of reference images and the entropy of error distribution would influence steganographic capacity, we carried out regression analysis. This section ends with an evaluation of steganographic rate-distortion performance.

4.1. Experimental Setup. The image samples for training and testing the MemNet were from the BOSSbase [72], which contains a collection of 10,000 greyscale photographs covering a wide variety of subjects and scenes. All the images were resampled to a resolution of 256×256 pixels through the Lanczos algorithm [73]. The number of convolution kernels per layer was configured to 64, the number of total memory cells was configured to 3, and the number of residual units per cell was configured to 3. The kernel weights were initialised by the Xavier initialisation [74]. The model was trained on 8,000 images over 100 epochs by the Adam optimiser [75] with an initial learning rate set to 10^{-3} and scheduled to decay exponentially after every epoch. Large-scale assessments were conducted on 2,000 test images. The inference process was simulated on selected standard test images from the USC-SIPI database [76].

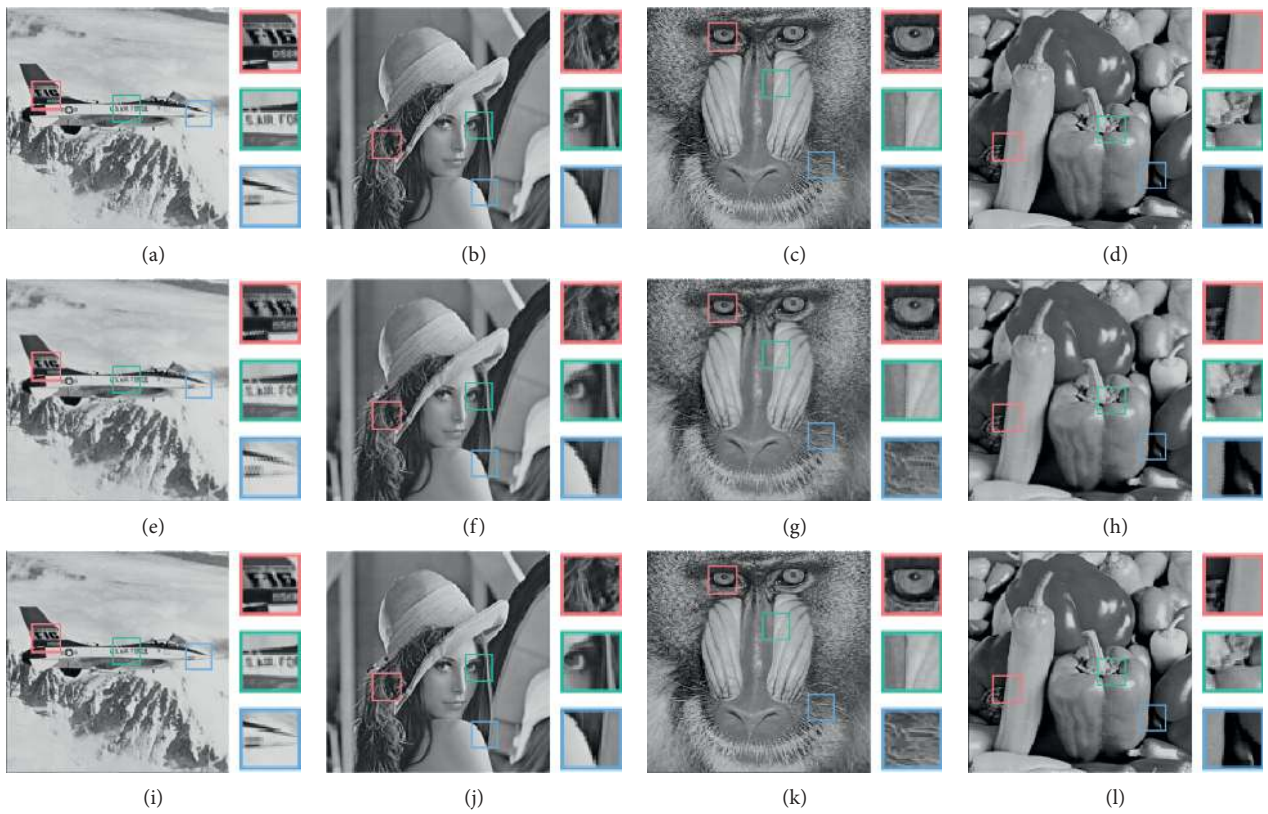
4.2. Visual Quality Analysis. Starting from Figures 5 and 6, we can catch a glimpse of the extent to which the model can refine the pre-processed images. It can be observed that the visual quality of posterior images is better than that of prior images, especially at the edges and in textural areas. The same outcome is reflected in the peak signal-to-noise ratio (PSNR) of images, measured in decibel (dB). Results suggest that the neural network model indeed has a stronger ability to model nonlinearity and complex pattern.

4.3. Entropy Analysis. Figure 7 shows that the posterior error distribution is more concentrated and its entropy smaller, whereas the prior error distribution is comparatively more diffuse. However, it is striking that the height of the peak bin (usually h_0) on the posterior histogram is not always higher than the height of the same bin on the prior histogram. A possible explanation would be that some image samples contain a relatively large number of smooth patches on which a naïve linear predictor may perform sufficiently well.

4.4. Cumulative Frequency Analysis. In order to better understand how the prior and posterior prediction errors distribute, we analyse their cumulative frequencies. Figure 8 presents cumulative distribution function (CDF) plots, where the 95$^{\text{th}}$ percentile gives the maximum error value below which 95% of errors fall. It is evident that the rate of convergence of the posterior error distribution is faster than that of the prior error distribution, confirming again that posterior errors are more concentrated and the magnitude of these is smaller on average.

4.5. Large-Scale Assessment. In addition to evaluating the performance on individual selected images from the USC-SIP database, we provide a large-scale assessment based on a large number of test samples from the BOSSbase. Figure 9(a) depicts the probability density of PSNRs of prior and posterior images. Figure 9(b) shows the probability density of entropies of prior and posterior error distributions. Figure 9(c) reveals the average rates of convergence of prior and posterior errors. On average, the visual quality of the posterior errors is higher, the distribution of them is more peaked, and the convergence rate is faster.

4.6. Regression Analysis. While we have shown that our neural network model offers better visual quality and smaller entropy, it is still unclear how these factors may benefit

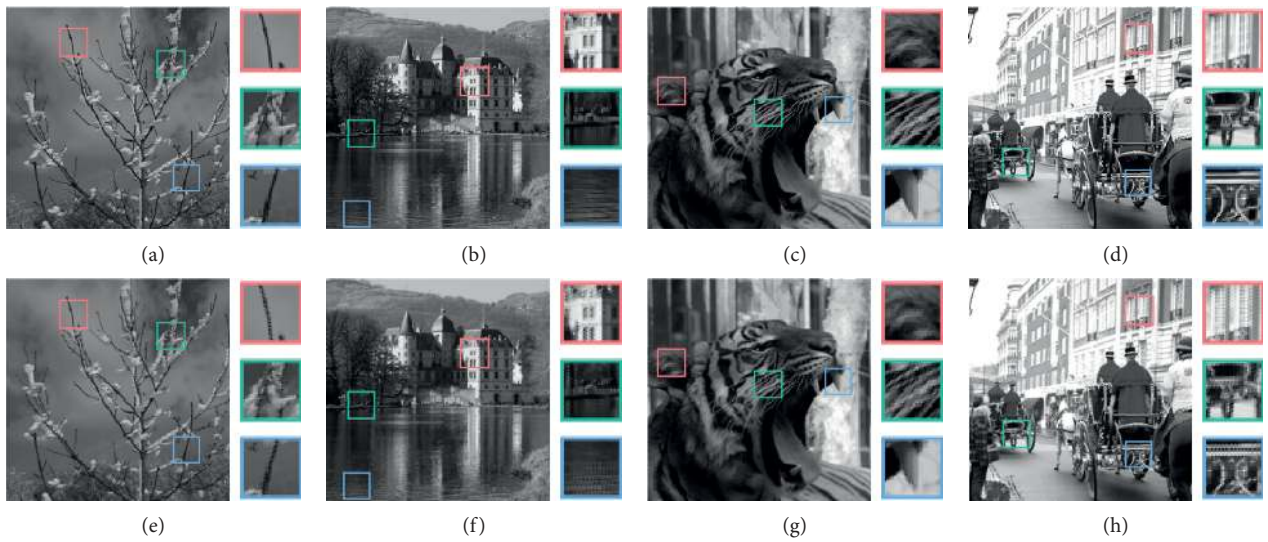


(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

Figure 5: Visual comparisons between the prior and posterior predictors based on images from the USC-SIPI database. Numerical data express the PSNRs. (a) Ground truth/aeroplane. (b) Ground truth/Lena. (c) Ground truth/mandrill. (d) Ground truth/peppers. (e) Prior/32.9055 dB. (f) Prior/34.5666 dB. (g) Prior/28.5752 dB. (h) Prior/35.3336 dB. (i) Posterior/36.3609 dB. (j) Posterior/37.6200 dB. (k) Posterior/29.3495 dB. (l) Posterior/39.9355 dB.



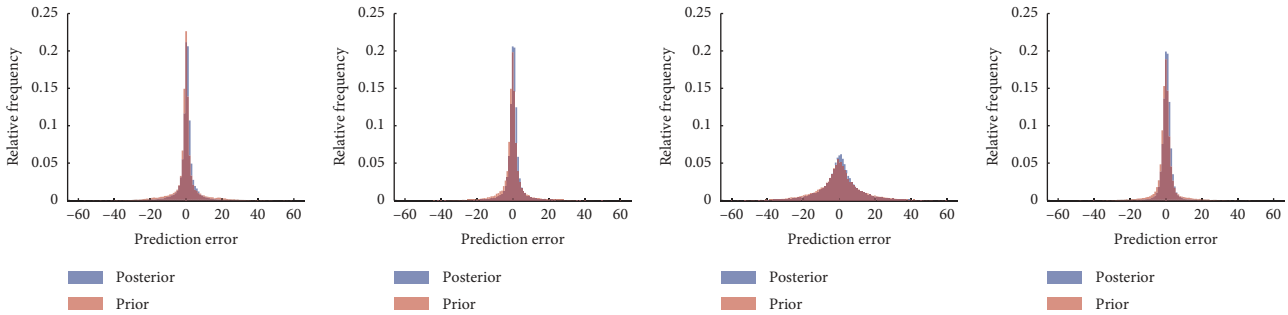
(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 6: Continued.

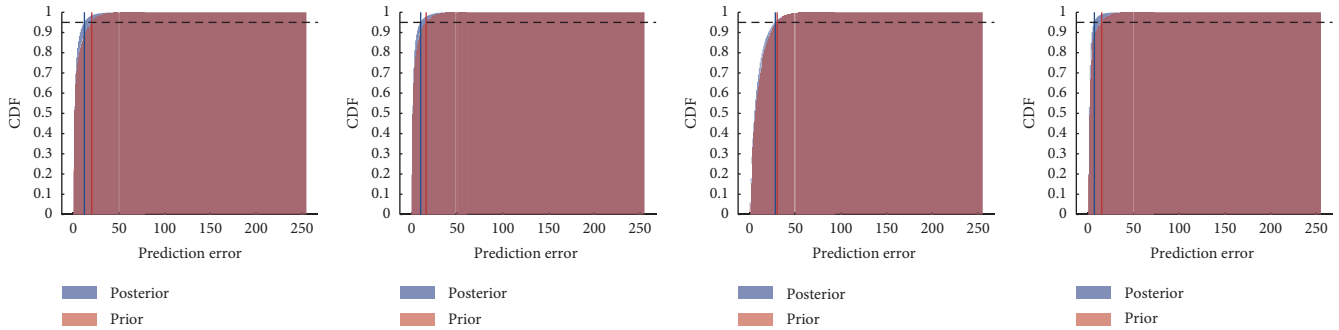
(i) (j) (k) (l)

Figure 6: Visual comparisons between the prior and posterior predictors based on images from the BOSSbase. Numerical data express the PSNRs. (a) Ground truth/0021. (b) Ground truth/0916. (c) Ground truth/1366. (d) Ground truth/1953. (e) Prior/31.5765 dB. (f) Prior/32.4795 dB. (g) Prior/34.9142 dB. (h) Prior/28.4180 dB. (i) Posterior/33.6192 dB. (j) Posterior/34.6770 dB. (k) Posterior/37.5122 dB. (l) Posterior/31.7238 dB.



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

Figure 7: Prediction error distributions. Numerical data express the entropy of distribution. (a) Aeroplane: prior/2.9529; posterior/2.6493. (b) Lena: prior/2.8683; posterior/2.5383. (c) Mandrill: prior/3.9392; posterior/3.8131. (d) Peppers: prior/2.7781; posterior/2.4074



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

Figure 8: Cumulative distribution function of prediction error distributions. Numerical data express the 95th percentile. (a) Aeroplane: prior/20; posterior/12. (b) Lena: prior/16; posterior/10. (c) Mandrill: prior/30; posterior/28. (d) Peppers: prior/15; posterior/7.

steganographic capacity. As a consequence, we carried out regression analysis amongst the PSNR of reference images, entropy of prediction errors, and maximum embedding rate, measured in bits per pixel (bpp). Figure 10 plots the results using the test samples from the BOSSbase with different threshold values θ which regulate the steganographic channel. As expected, the general trends suggest that the embedding rate is directly proportional to the PSNR of reference images and inversely proportional to the entropy of prediction errors.

4.7. Rate-Distortion Evaluation. We evaluate capacity and distortion by rate-distortion curves, as plotted in Figure 11. It can be observed that the maximum embedding rate increases with the increase of the threshold (the width of steganographic channel). The reason is straightforward: an increase in the threshold implies an increase in the number of bins for carrying the message. In addition to this, the observations suggest that the maximum embedding rate tends to be smaller for images containing more complex textures. It is because the prediction errors of such images

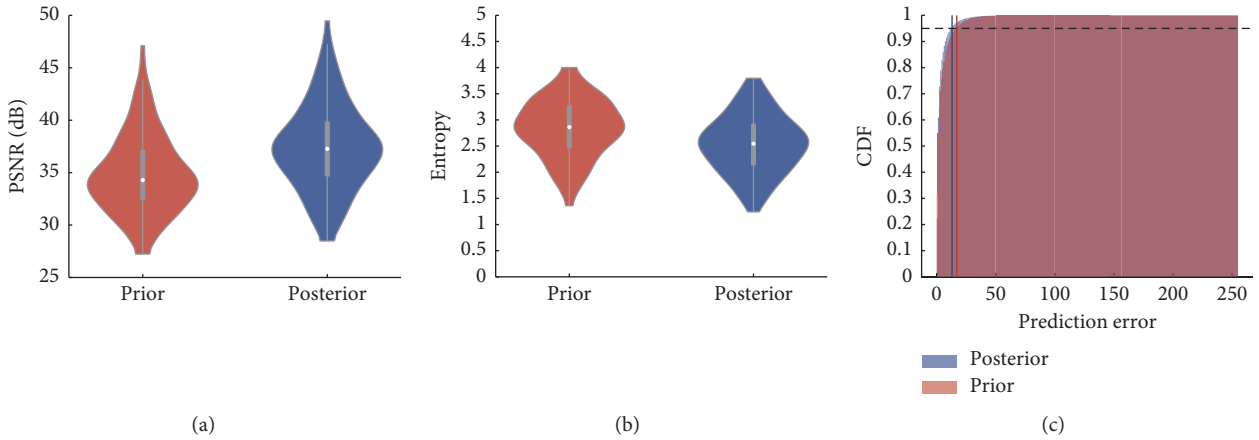


Figure 9: Statistical comparisons between prior and posterior predictors with respect to PSNRs of reference images, entropies of prediction error distributions, and cumulative distribution functions. (a) Prior/34.29 dB; posterior/37.27 dB. (b) Prior/2.863; posterior/2.546. (c) Prior/17; posterior/13.

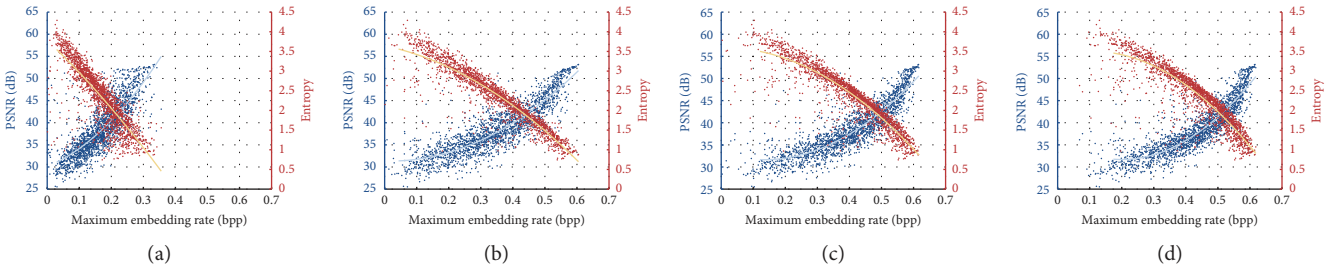


Figure 10: Regression analysis amongst PSNR of reference images, entropy of prediction errors, and maximum embedding rate at different threshold values that regulate the steganographic channel. (a) $\theta = 0$. (b) $\theta = 1$. (c) $\theta = 2$. (d) $\theta = 3$.

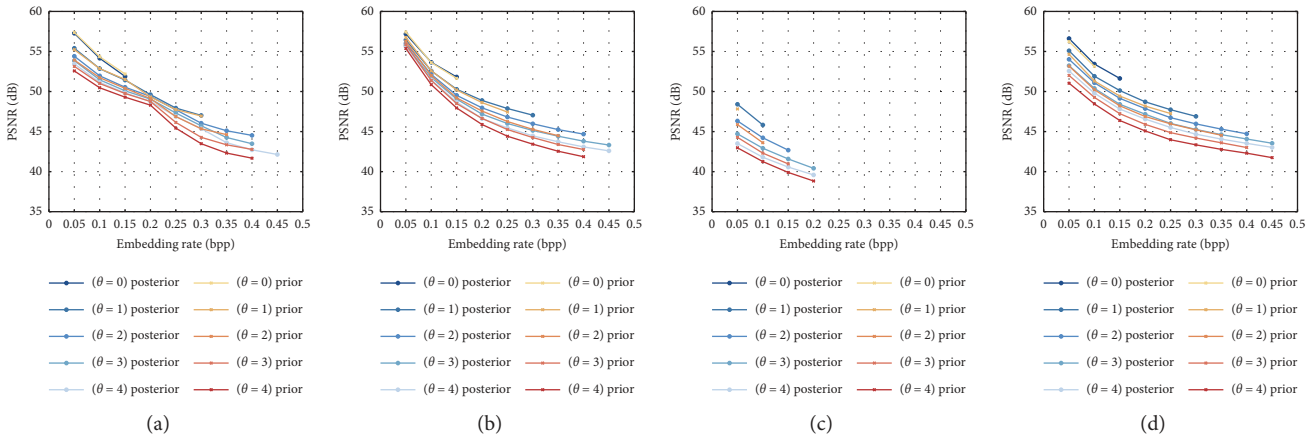


Figure 11: Rate distortion curves of applying prior and posterior predictors at different threshold values that regulate the steganographic channel. (a) Aeroplane. (b) Lena. (c) Mandrill. (d) Peppers.

are less concentrated and thus fewer bins are covered within the steganographic channel. There is a gradual and steady decline in the visual quality of stego images as embedding rate increases. The difference between the rate-distortion performances of the prior and posterior predictors is subtle for a small threshold value, but it becomes significant as the threshold value grows, with the posterior outperforming the prior. The underlying explanation for the trend may be that the naïve predictor and the neural network model have similar abilities to estimate smooth patches, for which both methods can often estimate perfectly. Nonetheless, the latter excels over the former when estimating textural patches, for which neither methods can offer accurate prediction but the neural network gives smaller error magnitude in general. Figure 12 lists stego images generated by embedding a simulated message into the cover images. The intended

Figure 12: Stego images generated by embedding a random bitstream with different threshold values that regulate the steganographic channel. Numerical data express the PSNRs and maximum embedding rates. (a) $\theta = 0$, 51.52 dB, 0.1652 bpp. (b) $\theta = 0$, 51.51 dB, 0.1607 bpp. (c) $\theta = 0$, 51.29 dB, 0.0470 bpp. (d) $\theta = 0$, 51.50 dB, 0.1554 bpp. (e) $\theta = 1$, 46.72 dB, 0.3240 bpp. (f) $\theta = 1$, 46.71 dB, 0.3248 bpp. (g) $\theta = 1$, 45.62 dB, 0.1057 bpp. (h) $\theta = 1$, 46.68 dB, 0.3191 bpp. (i) $\theta = 2$, 44.46 dB, 0.4039 bpp. (j) $\theta = 2$, 44.53 dB, 0.4150 bpp. (k) $\theta = 2$, 42.48 dB, 0.1581 bpp. (l) $\theta = 2$, 44.53 dB, 0.4210 bpp.

message is often assumed to have been compressed and encrypted and thus can be reasonably simulated by a random bit stream and a random trit stream.

5. Conclusions

This paper studies a neural analytics module compatible with the HS coding module. We propose a novel prediction mechanism which follows a two-step pipeline: first a pre-estimated image is generated by a conventional linear predictor and then the prior estimation is refined by an LSTM-based vision model called the MemNet. It is believed that this neural network model is to some extent biologically plausible and we have validated the effectiveness of the model for refining the prior estimation in terms of the visual quality and the entropy of error distribution. Furthermore, the impact of refinement to steganographic capacity has been analysed and a better rate-distortion performance was achieved. We envision that by joining this neural analytics module with a state-of-the-art HS coding module, the steganographic performance can be further improved. It is also interesting to investigate the possibility of combining different pre-processing predictors and post-processing neural network models to achieve a higher prediction accuracy. We hope this paper can prove instructive for future research on reversible steganography with deep learning.

Data Availability

The data and source code used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this paper.

References

[1] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding-a survey," *Proceedings of the the the the the IEEE*, vol. 87, no. 7, pp. 1062–1078, 1999.

[2] I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing*, vol. 6, no. 12, pp. 1673–1687, 1997.

[3] R. Liu and T. Tan, "An SVD-based watermarking scheme for protecting rightful ownership," *IEEE Transactions on Multimedia*, vol. 4, no. 1, pp. 121–128, 2002.

[4] C.-C. Lai and C.-C. Tsai, "Digital image watermarking using discrete wavelet transform and singular value decomposition," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 11, pp. 3060–3063, 2010.

[5] J. Fridrich, "Image watermarking for tamper detection," in *Proceedings of International Conference on Image Processing (ICIP)*, pp. 404–408, Chicago, IL, USA, October 1998.

[6] D. Kundur and D. Hatzinakos, "Digital watermarking for telltale tamper proofing and authentication," *Proceedings of the the the the the IEEE*, vol. 87, no. 7, pp. 1167–1180, 1999.

[7] T.-Y. Lee and S. D. Lin, "Dual watermark for image tamper detection and recovery," *Pattern Recognition*, vol. 41, no. 11, pp. 3497–3506, 2008.

[8] J. Fridrich, M. Goljan, P. Lisonek, and D. Soukal, "Writing on wet paper," *IEEE Transactions on Signal Processing*, vol. 53, no. 10, pp. 3923–3935, 2005.

[9] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," *EURASIP Journal on Information Security*, vol. 2014, no. 1, pp. 1–13, 2014.

[10] C.-Y. Chang and S. Clark, "Practical linguistic steganography using contextual synonym substitution and a novel vertex coding method," *Computational Linguistics*, vol. 40, no. 2, pp. 403–448, 2014.

[11] C. Szegedy, W. Zaremba, I. Sutskever et al., "Intriguing properties of neural networks," in *Proceedings of International Conference on Learning Representations (ICLR)*, pp. 1–10, Banff, Canada, April 2014.

[12] J. Fridrich, M. Goljan, and R. Du, "Invertible authentication," in *Proceedings of the the the the SPIE*, vol. 4314, pp. 197–208, San Jose, CA, USA, January 2001.

[13] C. De Vleeschouwer, J.-F. Delaigle, and B. Macq, "Circular interpretation of bijective transformations in lossless watermarking for media asset management," *IEEE Transactions on Multimedia*, vol. 5, no. 1, pp. 97–105, 2003.

[14] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 8, pp. 890–896, 2003.

[15] A. M. Alattar, "Reversible watermark using the difference expansion of a generalized integer transform," *IEEE Transactions on Image Processing*, vol. 13, no. 8, pp. 1147–1156, 2004.

[16] M. U. Celik, G. Sharma, and A. M. Tekalp, "Lossless watermarking for image authentication: a new framework and an implementation," *IEEE Transactions on Image Processing*, vol. 15, no. 4, pp. 1042–1049, 2006.

[17] S. Lee, C. D. Yoo, and T. Kalker, "Reversible image watermarking based on integer-to-integer wavelet transform," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 3, pp. 321–330, 2007.

[18] X. Huang, A. Nishimura, and I. Echizen, "A reversible acoustic steganography for integrity verification," in *Proceedings of International Workshop on Digital Watermarking (IWDW)*, pp. 305–316, Seoul, Korea, October 2010.

[19] D. Coltuc, "Low distortion transform for reversible watermarking," *IEEE Transactions on Image Processing*, vol. 21, no. 1, pp. 412–417, 2012.

[20] W. Zhang, X. Hu, X. Li, and Y. Nenghai, "Optimal transition probability of reversible data hiding for general distortion metrics and its applications," *IEEE Transactions on Image Processing*, vol. 24, no. 1, pp. 294–304, 2015.

[21] B. Ma and Y. Q. Shi, "A reversible data hiding scheme based on code division multiplexing," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 9, pp. 1914–1927, 2016.

[22] Y.-Q. Shi, X. Li, X. Zhang, H.-T. Wu, and B. Ma, "Reversible data hiding: advances in the past two decades," *IEEE Access*, vol. 4, pp. 3210–3237, 2016.

[23] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[24] W. Tang, S. Tan, B. Li, and J. Huang, "Automatic steganographic distortion learning using a generative adversarial network," *IEEE Signal Processing Letters*, vol. 24, no. 10, pp. 1547–1551, 2017.

[25] J. Hayes and G. Danezis, "Generating steganographic images via adversarial training," in *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1954–1963, Long Beach, CA, USA, December 2017.

[26] W. Tang, B. Li, S. Tan, M. Barni, and J. Huang, "CNN-based adversarial embedding for image steganography," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 8, pp. 2074–2087, 2019.

[27] L. Zhou, G. Feng, L. Shen, and X. Zhang, "On security enhancement of steganography via generative adversarial image," *IEEE Signal Processing Letters*, vol. 27, pp. 166–170, 2019.

[28] J. Yang, D. Ruan, J. Huang, X. Kang, and Y.-Q. Shi, "An embedding cost learning framework using GAN," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 839–851, 2020.

[29] J. Liu, Y. Ke, Z. Zhang et al., "Recent advances of image steganography with generative adversarial networks," *IEEE Access*, vol. 8, pp. 60 575–660 597, 2020.

[30] C.-C. Chang, "Adversarial learning for invertible steganography," *IEEE Access*, vol. 8, pp. 425–435, 2020.

[31] J. Fridrich, M. Goljan, and R. Du, "Lossless data embedding–new paradigm in digital watermarking," *EURASIP Journal on Advances in Signal Processing*, vol. 2002, pp. 185–196, Article ID 986842, 2002.

[32] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5967–5976, Honolulu, HI, USA, July 2017.

[33] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 354–362, 2006.

[34] G. Coatrieux, W. Wei Pan, N. Cuppens-Boulahia, F. Cuppens, and C. Roux, "Reversible watermarking based on invariant image classification and dynamic histogram shifting," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 111–120, 2013.

[35] X. Li, B. Li, B. Yang, and T. Zeng, "General framework to histogram-shifting-based reversible data hiding," *IEEE Transactions on Image Processing*, vol. 22, no. 6, pp. 2181–2191, 2013.

[36] X. Hu, W. Zhang, X. Li, and N. Yu, "Minimum rate prediction and optimized histograms modification for reversible data

hiding," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 3, pp. 653–664, 2015.

[37] X. Li, W. Zhang, X. Gui, and B. Yang, "Efficient reversible data hiding based on multiple histograms modification," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 2016–2027, 2015.

[38] J. Wang, J. Ni, X. Zhang, and Y. Q. Shi, "Rate and distortion optimization for reversible data hiding using multiple histogram shifting," *IEEE Transactions on Cybernetics*, vol. 47, no. 2, pp. 315–326, 2017.

[39] J. Wang, X. Chen, J. Ni, N. Mao, and Y. Shi, "Multiple histograms-based reversible data hiding: framework and realization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 8, pp. 2313–2328, 2020.

[40] W. Qi, X. Li, T. Zhang, and Z. Guo, "Optimal reversible data hiding scheme based on multiple histograms modification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 8, pp. 2300–2312, 2020.

[41] D. M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 721–730, 2007.

[42] M. Fallahpour, "Reversible image data hiding based on gradient adjusted prediction," *IEICE Electronics Express*, vol. 5, no. 20, pp. 870–876, 2008.

[43] C.-C. Lin, W.-L. Tai, and C.-C. Chang, "Multilevel reversible data hiding based on histogram modification of difference images," *Pattern Recognition*, vol. 41, no. 12, pp. 3582–3591, 2008.

[44] X. Li, B. Yang, and T. Zeng, "Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection," *IEEE Transactions on Image Processing: A Publication of the IEEE Signal Processing Society*, vol. 20, no. 12, pp. 3524–3533, 2011.

[45] I.-C. Dragoi and D. Coltuc, "Local-prediction-based difference expansion reversible watermarking," *IEEE Transactions on Image Processing*, vol. 23, no. 4, pp. 1779–1790, 2014.

[46] H. J. Hwang, S. Kim, and H. J. Kim, "Reversible data hiding using least square predictor via the LASSO," *EURASIP Journal on Image and Video Processing*, vol. 1, p. 42, 2016.

[47] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, 2016.

[48] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: residual learning of deep CNN for image denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[49] C. Ledig, L. Theis, F. Huszár et al., "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 105–114, Honolulu, HI, USA, July 2017.

[50] W. Lai, J. Huang, N. Ahuja, and M. Yang, "Deep Laplacian pyramid networks for fast and accurate super-resolution," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5835–5843, Honolulu, HI, USA, July 2017.

[51] J. Chen, J. Chen, H. Chao, and M. Yang, "Image blind denoising with generative adversarial network based noise modeling," in *Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3155–3164, Salt Lake City, UT, USA, June 2018.

[52] V. Lempitsky, A. Vedaldi, and D. Ulyanov, "Deep image prior," in *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9446–9454, Salt Lake City, UT, USA, June 2018.

[53] S. Anwar, S. Khan, and N. Barnes, "A deep journey into super-resolution: a survey," *ACM Computing Survey*, vol. 53, no. 3, 2020.

[54] Y. Tai, J. Yang, X. Liu, and C. Xu, "MemNet: a persistent memory network for image restoration," in *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pp. 4549–4557, Venice, Italy, October 2017.

[55] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[56] E. B. Wilson, "First and second laws of error," *Journal of the American Statistical Association*, vol. 18, no. 143, pp. 841–851, 1923.

[57] V. Sachnev, H. J. Hyoung Joong Kim, J. Jeho Nam, S. Suresh, and Y.-Q. Yun Qing Shi, "Reversible watermarking algorithm using sorting and prediction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 7, pp. 989–999, 2009.

[58] D. Hassabis, D. Kumaran, C. Summerfield, and M. Botvinick, "Neuroscience-inspired artificial intelligence," *Neuron*, vol. 95, no. 2, pp. 245–258, 2017.

[59] P. Dayan and L. F. Abbott, *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*, The MIT Press, Cambridge, MA, USA, 2005.

[60] J. Kirkpatrick, R. Pascanu, N. Rabinowitz et al., "Overcoming catastrophic forgetting in neural networks," *Proceedings of the the the the the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[61] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[62] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[63] Q. Liao and T. Poggio, "Bridging the gaps between residual learning, recurrent neural networks and visual cortex, MIT Center for Brains, Minds and Machines (CBMM)," 2016, https://arxiv.org/abs/1604.03640.

[64] M. Drozdzal, E. Vorontsov, G. Chartrand, S. Kadoury, and C. Pal, "The importance of skip connections in biomedical image segmentation," in *Proceedings of International Workshop on Deep Learning in Medical Image Analysis (DLMIA)*, pp. 179–187, Athens, Greece, October 2016.

[65] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[66] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the the the the the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[67] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 25, pp. 1097–1105, 2012.

[68] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of International Conference on Learning Representations (ICLR)*, pp. 1–14, Diego, CA, USA, May 2015.

[69] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of International Conference on Machine Learning (ICML)*, pp. 448–456, Lille, France, July 2015.

[70] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of International Conference*

on Artificial Intelligence and Statistics (AISTATS), pp. 315–323, Fort Lauderdale, FL, USA, April 2011.

[71] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, and Dropout, "A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[72] P. Bas, T. Filler, and T. Pevný, "Break our steganographic system: the ins and outs of organizing BOSS," in *Proceedings of International Workshop on Information Hiding (IH)*, pp. 59–70, Prague, Czech Republic, May 2011.

[73] C. E. Duchon, "Lanczos filtering in one and two dimensions," *Journal of Applied Meteorology*, vol. 18, no. 8, pp. 1016–1022, 1979.

[74] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 249–256, Sardinia, Italy, May 2010.

[75] D. P. Kingma, J. Ba, and Adam, "A method for stochastic optimization," in *Proceedings of International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, May 2015.

[76] A. G. Weber, "The USC-SIPI image database: version 5," USC Viterbi School of Engineering, Signal and Image Processing Institute, Los Angeles, CA, USA, 2006.