SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

March 2023, Vol. 66 132305:1–132305:14
https://doi.org/10.1007/s11432-021-3453-5

Reinforcement learning of non-additive joint steganographic embedding costs with attention mechanism

Weixuan TANG[1], Bin LI[2,3*], Weixiang LI[2], Yuangen WANG[4] & Jiwu HUANG[2,3]

[1]*Institute of Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou* 510006*, China;*
[2]*Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen* 518060*, China;*
[3]*Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen* 518060*, China;*
[4]*School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou* 510006*, China*

Received 13 October 2021/Revised 21 January 2022/Accepted 8 March 2022/Published online 21 February 2023

Abstract Image steganography is the art and science of secure communication by concealing information within digital images. In recent years, the techniques of steganographic cost learning have developed rapidly. Although the existing methods can learn satisfactory additive costs, the interplay of different pixels' embedding impacts has not been considered, so the potential of learning may not be fully exploited. To overcome this limitation, in this paper, a reinforcement learning paradigm called JoPoL (joint policy learning) is proposed to extend the idea of additive cost learning to a non-additive situation. JoPoL aims to capture the interactions within pixel blocks by defining embedding policies and evaluating contributions of embedding impacts on a block level rather than a pixel level. Then, a policy network is utilized to learn optimal joint embedding policies for pixel blocks through interactions with the environment. Afterwards, these policies can be converted into joint embedding costs for practical message embedding. The structure of the policy network is designed with an effective attention mechanism and incorporated with the domain knowledge derived from traditional non-additive steganographic methods. The environment is responsible for assigning rewards according to the impacts of the sampled joint embedding actions, which are evaluated by the gradient information of a neural network-based steganalyzer. Experimental results show that the proposed non-additive method JoPoL significantly outperforms the existing additive methods against both feature-based and CNN-based steganalzyers over different payloads.

Keywords information hiding, non-additive steganography, steganalysis, cost learning, image processing

Citation Tang W X, Li B, Li W X, et al. Reinforcement learning of non-additive joint steganographic embedding costs with attention mechanism. Sci China Inf Sci, 2023, 66(3): 132305, https://doi.org/10.1007/s11432-021-3453-5

1 Introduction

Image steganography [1] can hide communication traces, and ensure communication security, so it has great research significance. Modern steganographic methods are designed under the distortion minimization framework [2]. Assuming that the embedding impacts of different pixels are independent, the distortion function can be simplified into an additive form, which is calculated as the sum of the modified pixels' embedding costs. Several effective cost functions have been proposed in the past decade. They can be roughly divided into three categories, including the functions based on heuristic principles [3–5], statistical models [6–8], and steganalytic detectors [9, 10]. However, these additive cost functions neglect the interplay of modifications, which may cause some traces detectable by modern steganalysis. Therefore, to improve security performance, some pioneer methods equipped with non-additive costs are proposed. The spatial non-additive methods, such as CMD (clustering modification directions) [11], SYNCH (synchronizing the selection channel) [12], and DeJoin (decomposing joint distortion) [13], are similar in synchronizing modification directions of adjacent pixels. In CMD and SYNCH, the cover image is decomposed into several sub-images, and a part of messages is embedded into each sub-image with the

* Corresponding author (email: libin@szu.edu.cn)

costs updated according to the modification directions of the embedded sub-images. In DeJoin, the joint embedding costs of neighboring pixels are directly calculated, and then decomposed into additive ones on different sub-images for practical embedding. As for JPEG images, BBC (block boundary continuity) [14] aims at restraining block artifacts by synchronizing or desynchronizing the modification directions of inter-block coefficients, while BBM (block boundary maintenance) [15] attempts to minimize the modifications on the spatial block boundaries by modifying a pair of intra-block coefficients. MCTSteg [16] is the first steganographic framework equipped with a reinforcement learning technique for non-additive distortion. In this method, MCTS (Monte Carlo tree search) is utilized to adjust the distortion distribution with sequential decisions, and a well-trained steganalyzer calculates the reward feedback for the adjusted costs. It has better security performance but requires a much longer embedding time. On the other side of the game, image steganalysis aims to detect the embedding traces within stego images. Modern steganalytic methods adopt supervised machine learning models. The hand-crafted feature-based steganalyzers utilize the ensemble classifier [17] or the linear classifier [18] to classify high-dimensional image statistical features [19–26]. In recent years, the steganalyzers based on CNN (convolutional neural network) have made tremendous progress, which benefits from the delicate network design and the domain knowledge inherited from hand-crafted feature-based steganalyzers [27–37].

With the rapid development of deep learning techniques, researchers also try to exploit deep learning to learn steganograhpic embedding costs. These methods can be divided into two categories for different application scopes. The first one is to learn additive embedding costs from scratch, which can be realized by GAN (generative adversarial network) [38, 39] or RL (reinforcement learning) [40, 41]. The deep networks try to learn the embedding change probabilities, which can be converted into embedding costs for practical message embedding. The other one is to adjust the pre-defined embedding costs, which can be realized by the adversarial examples technique [42, 43] or RL [16]. To asymmetrically adjust the embedding costs, either the gradients with respect to the input pixels are utilized to disable the target CNN steganalyzer [42, 43], or the optimally adjusted embedding costs are searched by MCTS [16].

Although deep learning techniques have been widely applied to learn additive embedding costs, learning non-additive embedding costs by neural networks has not been investigated. Note that MCTSteg learns to adjust predefined embedding costs, rather than learning costs from scratch. In this paper, JoPoL (joint policy learning) is proposed to extend the idea of additive cost learning to a non-additive case by capturing the interactions of embedding impacts. In JoPoL, an agent tries to learn optimal joint embedding policies for pixel blocks to maximize the joint rewards from an environment. To this end, a delicate network structure is designed. Inspired by traditional methods where non-additive costs are adjusted according to their additive versions, the policy network of JoPoL is equipped with an attention mechanism, wherein the attention maps are utilized to evaluate the embedding suitability to facilitate the learning of joint embedding policies. Specifically, joint embedding actions that better preserve the correlation within pixel blocks are paid more attention and encouraged. For the environment, pixel-level rewards within a pixel block are aggregated into joint rewards, evaluating the impacts of the overall actions on deceiving a steganalyzer. The learned joint embedding policies can be converted into joint embedding costs, which can be decomposed into additive costs for practical message embedding.

The contributions of this work are summarized as follows.

• A novel non-additive cost learning paradigm called JoPoL is developed, which considers the interplay of embedding impacts within pixel blocks. To the best of our knowledge, this is the first work that tries to automatically learn non-additive embedding costs from scratch by neural networks.

• The attention mechanism is introduced into cost learning for the first time. Such a mechanism can utilize both the domain knowledge derived from traditional non-additive methods and the learning ability of deep learning models. The learned attention maps can be used to improve the initial joint embedding policies to differentiate the suitability of different joint embedding actions. Besides, attention maps can be shared for symmetric embedding patterns so as to reduce learning complexity.

• Experimental results show that the non-additive JoPoL significantly outperforms the existing additive methods against different steganalyzers. Meanwhile, extensive analysis demonstrates that the data-driven JoPoL can automatically balance the tradeoff between enhancing the effect of concentrating modification directions and increasing the change rate.

The rest of this paper is organized as follows. The background of traditional non-additive steganographic methods and additive cost learning methods are given in Section 2. The proposed JoPoL is described in Section 3. Experimental results are presented in Section 4. Conclusion is drawn in Section 5.

2 Fundamentals and background

2.1 Notations

In this paper, capital letters in bold represent matrices, lowercase letters in bold represent vectors, and the corresponding lowercase letters represent their elements. Specifically, cover and stego images are respectively denoted as $\boldsymbol{X} = (x_{i,j})^{H\times W}$ and $\boldsymbol{Y} = (y_{i,j})^{H\times W}$, where H and W are the height and width of the image, $1 \leqslant i \leqslant H$, and $1 \leqslant j \leqslant W$. The cover images can also be represented in a 2×2 block-level manner as $\boldsymbol{U} = (\boldsymbol{u}_{a,b})^{H/2\times W/2}$ and 1×2 block-level manner as $\boldsymbol{V} = (\boldsymbol{v}_{a,b})^{H\times W/2}$, where $\boldsymbol{u}_{a,b} = (x_{i,j}, x_{i,j+1}, x_{i+1,j}, x_{i+1,j+1})$ ($i = (a-1)\times 2+1$ and $j = (b-1)\times 2+1$) and $\boldsymbol{v}_{a,b} = (x_{i,j}, x_{i,j+1})$ ($i = a$ and $j = (b-1)\times 2+1$). The set of embedding operation is denoted as I, where $I = \{+1, -1, 0\}$ for ternary embedding.

2.2 Distortion minimization framework

The distortion minimization framework formulates the message embedding process as a constrained optimization problem as

$$\min_{\boldsymbol{Y}} D(\boldsymbol{X}, \boldsymbol{Y}), \quad \text{s.t. } \psi(\boldsymbol{Y}) = C, \tag{1}$$

where $D(\boldsymbol{X}, \boldsymbol{Y})$ is the distortion of changing $\boldsymbol{X}$ into $\boldsymbol{Y}$, $\psi(\boldsymbol{Y})$ is the message payload carried by $\boldsymbol{Y}$, and C is the target amount of message payload.

The distortion can be simplified as an additive form. The embedding impacts introduced by different pixels are assumed to be independent, and the distortion can be calculated as the sum of embedding costs of modified pixels as

$$D(\boldsymbol{X}, \boldsymbol{Y}) = \sum_{i=1}^{H}\sum_{j=1}^{W} \rho_{i,j}^{m_{i,j}}\delta \ \ (m_{i,j} \neq 0), \tag{2}$$

where $\rho_{i,j}^{m_{i,j}}$ is the embedding cost of modifying $x_{i,j}$ into $x_{i,j} + m_{i,j}$, and $\delta(z)$ is the indicator function as

$$\delta(z) = \begin{cases} 1, & \text{if } z \text{ is true,} \\ 0, & \text{if } z \text{ is false.} \end{cases} \tag{3}$$

The distortion can also be defined as a non-additive joint form, similar to that in DeJoin [13]. For example, the distortion can be defined as the sum of the joint embedding costs for 1×2 pixel blocks as

$$D(\boldsymbol{X}, \boldsymbol{Y}) = \sum_{a=1}^{H}\sum_{b=1}^{W/2} \rho_{a,b}^{\boldsymbol{n}_{a,b}}\delta \ \ (\boldsymbol{n}_{a,b} \neq (0,0)), \tag{4}$$

where $\boldsymbol{n}_{a,b} = (m_{i,j}, m_{i,j+1})$, and $\rho_{a,b}^{\boldsymbol{n}_{a,b}}$ is the cost of modifying $\boldsymbol{v}_{a,b} = (x_{i,j}, x_{i,j+1})$ into $\boldsymbol{v}_{a,b} + \boldsymbol{n}_{a,b} = (x_{i,j} + m_{i,j}, x_{i,j+1} + m_{i,j+1})$. The joint embedding change probabilities can be calculated as

$$p_{a,b}^{(m_1,m_2)} = \frac{\mathrm{e}^{-\lambda\rho_{a,b}^{(m_1,m_2)}}}{\sum_{(\tilde{m}_1,\tilde{m}_2)\in I^2}\mathrm{e}^{-\lambda\rho_{a,b}^{(\tilde{m}_1,\tilde{m}_2)}}}, \quad (m_1, m_2) \in I^2, \tag{5}$$

where λ is determined by the payload constraint. In the case of 1×2 pixel block, DeJoin decomposes the embedding process into two rounds. In the first round, the embedding change probabilities of the first pixels within the 1×2 pixel blocks can be calculated as the margin probabilities as

$$p_{i,j}^{m_1} = \sum_{m_2\in I} p_{a,b}^{(m_1,m_2)}, \quad m_1 \in I, \quad i = a, \quad j = 2\times b - 1. \tag{6}$$

In the second round, the embedding change probabilities of the second pixels within the 1×2 pixel blocks can be calculated as the conditional probabilities as

$$p_{i,j+1}^{m_2} = \frac{p_{a,b}^{(m_1,m_2)}}{p_{i,j}^{m_1}}, \quad m_2 \in I, \quad i = a, \quad j = 2\times b - 1. \tag{7}$$

In each round, the embedding change probabilities can be converted into embedding costs. Meanwhile, STC (Syndrome-Trellis codes) [2] and SPC (steganographic polar codes) [44] can be applied to message embedding.

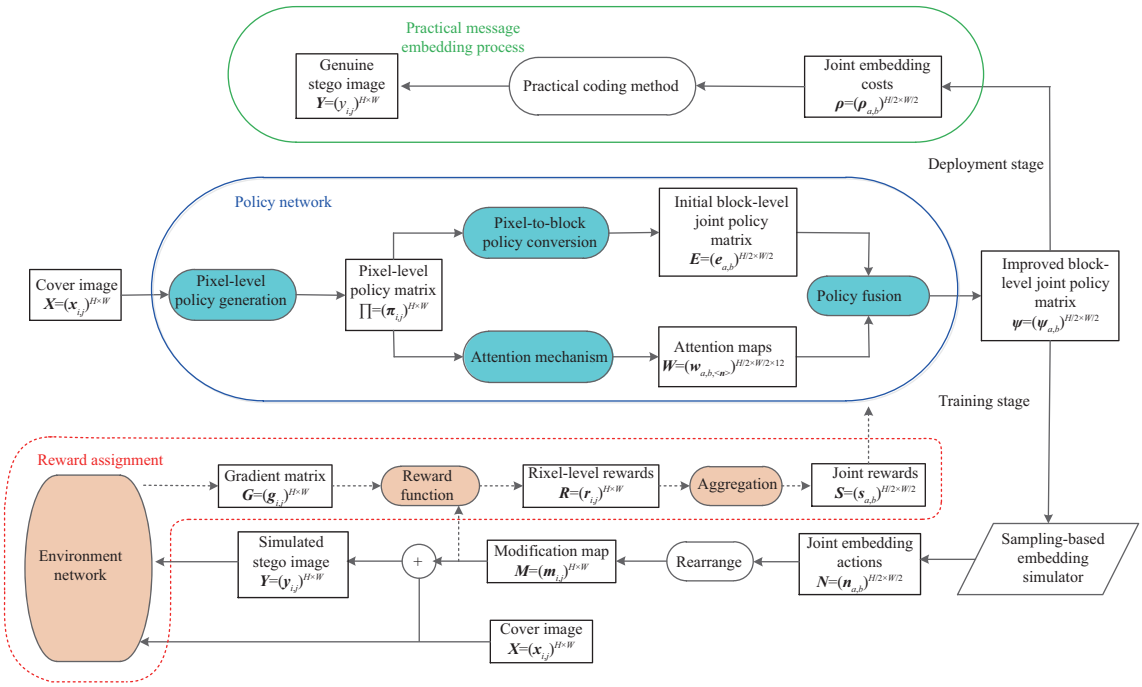


Figure 1 (Color online) Illustration of the proposed JoPoL paradigm.

2.3 Additive cost learning framework

ASDL-GAN (automatic steganographic distortion learning framework with generative adversarial network) [38] was the first cost learning framework that can learn additive embedding costs from scratch. In ASDL-GAN, the generator outputs the stego image according to the learned embedding change probabilities, while the discriminator distinguishes between the cover image and the stego image. A neural network-based embedding simulator is responsible for simulating message embedding in the feedforward process, and propagating significant gradients from the discriminator to the generator in the backpropagation process. UT-GAN (U-Net and double-tanh framework using GAN) [39] improves ASDL-GAN by utilizing more sophisticated network structures including U-Net and Double-Tanh unit. However, since the GAN-based methods adopt a neural network-based embedding simulator, it inevitably leads to the tradeoff between large propagated gradients and small modification deviation. To solve these problems, SPAR-RL (steganographic pixel-wise actions and rewards with reinforcement learning) [40] was proposed. In SPAR-RL, a policy network learns the optimal embedding policies, i.e., the embedding change probabilities, by maximizing the rewards assigned from an environment for the sampled actions, i.e., simulated embedding modifications. Meanwhile, a sampling-based embedding simulator is applied in SPAR-RL to generate accurate discrete modifications, and effective gradients can be propagated to the policy network. Considering the knowledge in the JPEG images, SPAR-RL could be extended to the discrete cosine transform (DCT) domain as JEC-RL (JPEG embedding cost with reinforcement learning) [41].

3 The proposed JoPoL paradigm

3.1 Paradigm overview

Under the RL framework, the idea of additive cost learning in SPAR-RL is extended to a non-additive situation, and a JoPoL paradigm is proposed to capture the interplay of embedding impacts within an $L\times L$ pixel block. This paper considers the case of $L=2$. In JoPoL, a policy network learns the optimal joint embedding policies with the maximum joint rewards assigned from an environment, as shown in Figure 1. Note traditional spatial steganographic methods improve additive cost functions into their non-additive versions by asymmetrically adjusting the embedding costs of different directions [11–13]. Such domain knowledge is incorporated with an attention mechanism into the design of the policy network. In recent years, the attention mechanism has been widely applied into different tasks, such as image

classification [45], image captioning [46], and machine translation [47]. Among them, the channel attention applied to image tasks [48–50] is most related to the attention mechanism in our proposed JoPoL. They have the similarity that the attention weights are used to distinguish different feature maps across the channel dimension. Compared to the existing channel attention mechanism wherein the attention map is learned implicitly, the attention mechanism in our proposed JoPoL is incorporated with the domain knowledge of non-additive steganography, and each attention map is explicitly associated with a block-level embedding pattern and is used to adjust its embedding probability.

The policy network consists of four modules, i.e., pixel-level policy generation, pixel-to-block policy conversion, attention mechanism, and policy fusion. The pixel-level policy generation takes the cover image $\boldsymbol{X}$ as input and outputs the pixel-level policy matrix $\boldsymbol{\Pi} = (\boldsymbol{\pi}_{i,j})^{H\times W}$ where $\pi_{i,j}(m)$ is the probability of taking action m for pixel $x_{i,j}$, and $m \in I$. Then, the three subsequent modules are used to improve the pixel-level policies into block-level policies with an attention mechanism. Specifically, the pixel-to-block policy conversion is responsible for tuning the pixel-level policy matrix $\boldsymbol{\Pi}$ into an initial block-level joint policy matrix $\boldsymbol{E} = (\boldsymbol{e}_{a,b})^{H/2\times W/2}$, where $e_{a,b}(\boldsymbol{n})$ is the joint probability of taking actions $\boldsymbol{n} = (m_1, m_2, m_3, m_4)$ for a pixel block $\boldsymbol{u}_{a,b} = (x_{i,j}, x_{i,j+1}, x_{i+1,j}, x_{i+1,j+1})$. This module does not consider the interactions of different modifications among the pixels within a block, so $e_{a,b}(\boldsymbol{n})$ is only the product of the pixel-level probabilities $\pi_{i,j}(m)$ within the block. In another branch, the attention mechanism aims to capture the interactions of embedding impacts within pixel blocks, and learn the block-level attention maps $\boldsymbol{W} = (w_{a,b,\langle \boldsymbol{n}\rangle})^{H/2\times W/2\times 12}$, wherein $w_{a,b,\langle \boldsymbol{n}\rangle}$ represents the adjustment weight for $e_{a,b}(\boldsymbol{n})$. Finally, to differentiate the suitability of different joint embedding actions, the policy fusion utilizes $\boldsymbol{W}$ to adjust $\boldsymbol{E}$ into the improved joint embedding policy matrix $\boldsymbol{\Psi} = (\boldsymbol{\psi}_{a,b})^{H/2\times W/2}$. Then, a sampling-based embedding simulator is adopted to sample joint embedding actions $\boldsymbol{N} = (\boldsymbol{n}_{a,b})^{H/2\times W/2}$ according to $\boldsymbol{\Psi}$, where $\boldsymbol{n}_{a,b} = (m_{i,j}, m_{i,j+1}, m_{i+1,j}, m_{i+1,j+1})$. $\boldsymbol{N}$ is then rearranged into a modification map $\boldsymbol{M} = (m_{i,j})^{H\times W}$, which is then added to the cover image $\boldsymbol{X}$ to obtain a simulated stego image $\boldsymbol{Y}$. On the other side, based on the gradient information of a CNN steganalyzer, an environment firstly assigns pixel-level rewards $\boldsymbol{R} = (r_{i,j})^{H\times W}$ for $\boldsymbol{M}$, which are then aggregated into the joint rewards $\boldsymbol{S} = (s_{a,b})^{H/2\times W/2}$ to indicate the impacts of joint embedding actions on deceiving the steganalyzer. The policy network can learn the optimal joint embedding policies $\boldsymbol{\Psi}$ by maximizing the joint rewards $\boldsymbol{S}$ from the environment. After training, the joint embedding policies can be inversely converted into joint embedding costs, and the decomposing technique in DeJoin can be applied to practical message embedding.

3.2 Obtaining joint policies via the policy network equipped with attention mechanism

The pixel-level policy generation is responsible for obtaining $\boldsymbol{\Pi} = (\boldsymbol{\pi}_{i,j})^{H\times W}$. It can be implemented by the existing additive steganographic method, where $\pi_{i,j}(m)$ is the embedding change probabilities of modifying $x_{i,j}$ into $x_{i,j}+m$. Then, the pixel-to-block policy conversion outputs the initial joint embedding policy matrix $\boldsymbol{E} = (\boldsymbol{e}_{a,b})^{H/2\times W/2}$. $e_{a,b}(\boldsymbol{n})$ is calculated as

$$e_{a,b}(\boldsymbol{n}) = \pi_{i,j}(m_1) \times \pi_{i,j+1}(m_2) \times \pi_{i+1,j}(m_3) \times \pi_{i+1,j+1}(m_4), \tag{8}$$

where $\boldsymbol{n} = (m_1, m_2, m_3, m_4)$, $i = (a-1)\times 2 + 1$, and $j = (b-1)\times 2 + 1$. Specifically, SPAR-RL-v2 is used in the pixel-level policy generation, and its network architecture is shown in Figure 2(a).

The attention mechanism takes the embedding change probability map $\boldsymbol{P} = (p_{i,j})^{H\times W}$ as input, where $p_{i,j} = (1-\pi_{i,j}(0))/2$, and turns it into attention maps by convolutions, where each map corresponds to a specific joint embedding action. In the case of a pixel-level action $m \in I$ and an $L\times L$ pixel block, a joint embedding action $\boldsymbol{n} \in I^{L\times L}$ is considered. Therefore, the number of attention maps grows exponentially with the increase of block size L, which brings a heavy computation burden and may make the network difficult to converge. Meanwhile, each joint embedding action has a specific embedding pattern, and some of the embedding patterns are symmetric in the sense that they can mutually turn into each other by taking the opposite sign, rotating, and flipping. For example, $(+1, -1, 0, +1)$ and $(-1, +1, 0, -1)$ are sign symmetric, $(0, +1, 0, +1)$ and $(+1, +1, 0, 0)$ are rotate symmetric, and $(+1, 0, 0, -1)$ and $(0, -1, +1, 0)$ are flip symmetric. In this way, 80 embedding patterns (there are $3^4 = 81$ embedding patterns in total, and the pattern (0,0,0,0) is excluded) can be divided into 12 groups, and the embedding patterns within each group are symmetric, as shown in Table 1. The symmetric embedding patterns have a similar first-order differentiable statistical property, and therefore they are assigned by the same attention map. In this way, the attention mechanism outputs 12 attention maps as $\boldsymbol{W} = (w_{a,b,\langle \boldsymbol{n}\rangle})^{H/2\times W/2\times 12}$, where $\langle \boldsymbol{n}\rangle$ is the

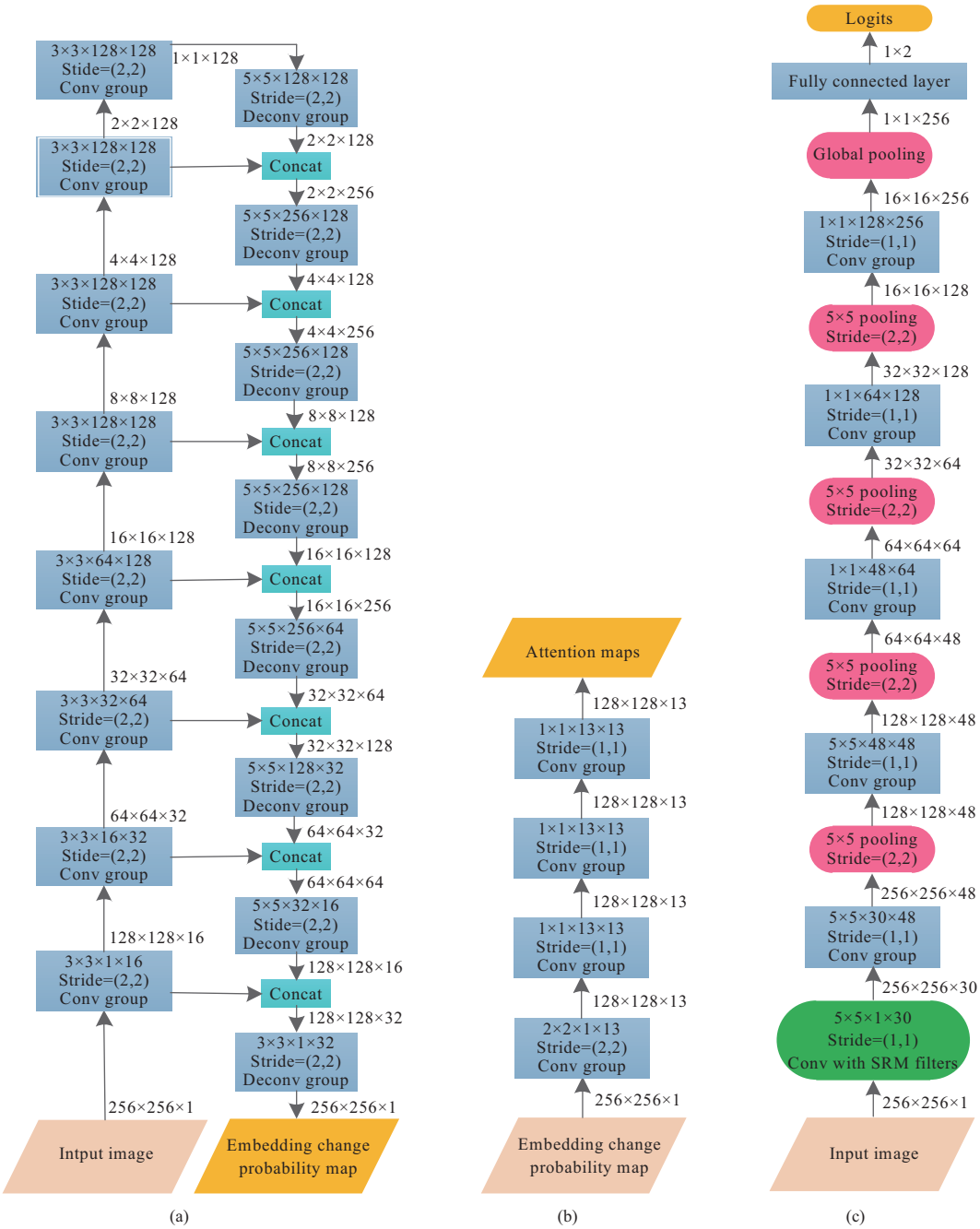


Figure 2 (Color online) The architecture of the learnable part in JoPoL, includes the pixel-level policy generation (a), the attention mechanism (b), and the environment network (c). Each convolutional or deconvolutional group is composed of a convolutional or deconvolutional layer, a batch normalization layer, and an activation function. The kernel configurations of the convolutional or deconvolutional layers are represented as: kernel height × kernel width × number of input feature maps × number of output feature maps. In the pixel-level policy generation, the ReLU activation function is applied in the first eight groups, while the leaky ReLU activation function is applied in the last eight groups. In the attention mechanism, the leaky ReLU activation function is applied in the first three groups, and the linear activation function is applied in the last group. In the environment network, the absolute operation is performed after the convolutional layer in the first group. The tanh activation function is applied in the first two groups, and the ReLU activation function is applied in the last three groups. The kernel configurations of the pooling layers are represented as: kernel height × kernel width. The sizes of the output feature maps are represented as: height × width × number of feature maps.

mapping from joint embedding action $\boldsymbol{n}$ to integer $z \in \{1, 2, \ldots, 12\}$, and $\boldsymbol{n} \in I^4$, $\boldsymbol{n} \neq (0, 0, 0, 0)$ and z is the index of the divided group. This module can be implemented as a CNN model with four groups of convolutions, and each group outputs 12 feature maps. The first group utilizes a 2×2 convolution to integrate the embedding change probabilities within the 2×2 pixel block. Then, the following three

Table 1 The mapping from joint embedding action $\boldsymbol{n}$ to integer $\langle \boldsymbol{n} \rangle$

$\boldsymbol{n}$	$\langle \boldsymbol{n} \rangle$
(0,0,0,+1), (0,0,+1,0), (0,+1,0,0), (+1,0,0,0), (0,0,0,−1), (0,0,−1,0), (0,−1,0,0), (−1,0,0,0)	1
(0,0,+1,−1), (0,0,−1,+1), (0,+1,0,−1), (0,−1,0,+1), (+1,0,−1,0), (+1,−1,0,0), (−1,0,+1,0), (−1,+1,0,0)	2
(0,0,+1,+1), (0,0,−1,−1), (0,+1,0,+1), (0,−1,0,−1), (+1,0,+1,0), (+1,+1,0,0), (−1,0,−1,0), (−1,−1,0,0)	3
(0,+1,−1,0), (0,−1,+1,0), (+1,0,0,−1), (−1,0,0,+1)	4
(0,+1,−1,+1), (0,+1,−1,−1), (0,−1,+1,+1), (0,−1,+1,−1), (+1,0,+1,−1) (+1,−1,0,−1), (+1,−1,+1,0), (−1,0,−1,+1), (−1,+1,0,+1), (−1,+1,−1,0) (+1,0,−1,−1), (+1,+1,0,−1), (+1,+1,−1,0), (−1,0,+1,+1), (−1,−1,0,+1), (−1,−1,+1,0)	5
(0,+1,+1,−1), (0,−1,−1,+1), (+1,0,−1,+1), (+1,−1,0,+1) (+1,−1,−1,0), (−1,0,+1,−1), (−1,+1,0,−1), (−1,+1,+1,0)	6
(0,+1,+1,0), (0,−1,−1,0), (+1,0,0,+1), (−1,0,0,−1)	7
(0,+1,+1,+1), (0,−1,−1,−1), (+1,0,+1,+1), (+1,+1,0,+1) (+1,+1,+1,0), (−1,0,−1,−1), (−1,−1,0,−1), (−1,−1,−1,0)	8
(+1,−1,−1,+1), (−1,+1,+1,−1)	9
(+1,+1,−1,−1), (+1,−1,+1,−1), (−1,+1,−1,+1), (−1,−1,+1,+1)	10
(+1,+1,+1,−1), (+1,+1,−1,+1), (+1,−1,+1,+1), (+1,−1,−1,−1) (−1,+1,+1,+1), (−1,+1,−1,−1), (−1,−1,+1,−1), (−1,−1,−1,+1)	11
(+1,+1,+1,+1), (−1,−1,−1,−1)	12

groups all utilize a 1×1 convolution to learn the attention weight. Sigmoid function is applied at the end of the last group to restrict $w_{a,b,\langle \boldsymbol{n} \rangle} \in (0,1)$, where $w_{a,b,\langle \boldsymbol{n} \rangle}$ is the attention weight for $e_{a,b}(\boldsymbol{n})$. The network architecture of the attention mechanism is illustrated in Figure 2(b).

The policy fusion is responsible for fusing the initial joint embedding policy matrix $\boldsymbol{E} = (\boldsymbol{e}_{a,b})^{H/2 \times W/2}$ and the attention maps $\boldsymbol{W} = (w_{a,b,\langle \boldsymbol{n} \rangle})^{H/2 \times W/2 \times 12}$ into the improved joint embedding policy matrix $\boldsymbol{\Psi} = (\boldsymbol{\psi}_{a,b})^{H/2 \times W/2}$. It can be seen from (16) that the joint embedding cost can be calculated as the logarithmic ratio between $\psi_{a,b}(\boldsymbol{n})$ where $\boldsymbol{n} = (0,0,0,0)$ and $\psi_{a,b}(\boldsymbol{n})$ where $\boldsymbol{n} \in I^4$. To stablize the learning process, $\psi_{a,b}(\boldsymbol{n})$ is fixed to $e_{a,b}(\boldsymbol{n})$ when $\boldsymbol{n} = (0,0,0,0)$, and $\psi_{a,b}(\boldsymbol{n})$ can be specifically learned for other 80 joint embedding actions. The $\psi_{a,b}(\boldsymbol{n})$ for other 80 joint embedding actions is the product between the initial joint probability $e_{a,b}(\boldsymbol{n})$ and the attention weight $w_{a,b,\langle \boldsymbol{n} \rangle}$. Then it is normalized afterwards so that the summation of the probabilities of 81 joint embedding actions equals to 1, as shown in

$$\psi_{a,b}(\boldsymbol{n}) = \begin{cases} e_{a,b}(\boldsymbol{n}), & \text{if} \quad \boldsymbol{n} = \boldsymbol{n}_0, \\ \dfrac{e_{a,b}(\boldsymbol{n}) \times w_{a,b,\langle \boldsymbol{n} \rangle}}{\sum_{\boldsymbol{n} \in I^4, \boldsymbol{n} \neq \boldsymbol{n}_0} e_{a,b}(\boldsymbol{n}) \times w_{a,b,\langle \boldsymbol{n} \rangle}} \times \big(1 - e_{a,b}(\boldsymbol{n}_0)\big), & \text{otherwise}, \end{cases} \tag{9}$$

where $\boldsymbol{n}_0 = (0,0,0,0)$.

3.3 Assigning joint reward from the environment

To facilitate effective non-additive cost learning, the environment assigns joint rewards $\boldsymbol{S} = (s_{a,b})^{H/2 \times W/2}$ to the joint embedding actions $\boldsymbol{N} = (\boldsymbol{n}_{a,b})^{H/2 \times W/2}$ sampled by the agent, where $s_{a,b}$ evaluates the impact of $\boldsymbol{n}_{a,b}$ on deceiving the steganalyzer. In this paper, a joint reward assignment is proposed based on the gradient information of an environment network. Our environment network is an enhanced version of Xu-Net [31], where the preprocessing layer utilizes 30 high-pass filters, and the width of the first to the fifth layer is 48, 48, 64, 128, and 256, respectively. The network architecture of the environment network is illustrated in Figure 2(c). Using the environment network, a gradient matrix $\boldsymbol{G} = (g_{i,j})^{H \times W}$ can be calculated, where $g_{i,j}$ is the gradient of the environment network's cross-entropy loss with respect to modification $m_{i,j}$. Then, the reward matrix $\boldsymbol{R} = (r_{i,j})^{H \times W}$ is obtained by $\boldsymbol{M} = (m_{i,j})^{H \times W}$ and $\boldsymbol{G} = (g_{i,j})^{H \times W}$ as

$$r_{i,j} = \xi \times \operatorname{sign}(m_{i,j}) \times g_{i,j}, \tag{10}$$

where ξ is a constant controlling the reward magnitude. Finally, the pixel-level rewards $r_{i,j}$ within the pixel block is accumulated into the joint reward $s_{a,b}$ as

$$s_{a,b} = r_{i,j} + r_{i,j+1} + r_{i+1,j} + r_{i+1,j+1}. \tag{11}$$

In this way, $s_{a,b}$ can measure the impact of the joint embedding action. The larger $s_{a,b}$, the more likely the simulated stego image will be misclassified by the CNN-based steganalyzer.

3.4 The implementation of JoPoL

To implement JoPoL, the policy network and the environment network are trained to learn the optimal joint embedding policies, and then they are converted into embedding costs for message embedding. The training process contains multiple iterations, and the steps in one specific iteration are given as follows.

(1) The policy network takes the cover image $\boldsymbol{X} = (x_{i,j})^{H\times W}$ as input, and outputs the improved joint policy matrix $\boldsymbol{\Psi} = (\boldsymbol{\psi}_{a,b})^{H/2\times W/2}$.

(2) The sampling-based embedding simulator is exploited to sample joint embedding actions $\boldsymbol{N} = (\boldsymbol{n}_{a,b})^{H/2\times W/2}$ according to $\boldsymbol{\Psi} = (\boldsymbol{\psi}_{a,b})^{H/2\times W/2}$, which can be used to form modification map $\boldsymbol{M} = (m_{i,j})^{H\times W}$. Then, the modification map $\boldsymbol{M} = (m_{i,j})^{H\times W}$ is added to the cover image $\boldsymbol{X} = (x_{i,j})^{H\times W}$ to generate a simulated stego image $\boldsymbol{Y} = (y_{i,j})^{H\times W}$.

(3) The environment assigns joint rewards $\boldsymbol{S} = (s_{a,b})^{H/2\times W/2}$ to the sampled joint embedding actions $\boldsymbol{N} = (\boldsymbol{n}_{a,b})^{H/2\times W/2}$ according to (10) and (11).

(4) Update the parameters of the attention mechanism in the policy network by minimizing the total loss l_A, which is the weighted sum of the reward loss l_R and the capacity loss l_C as

$$l_A = \alpha \cdot l_R + \beta \cdot l_C, \tag{12}$$

$$l_R = -\frac{4}{H\times W}\sum_{a=1}^{H/2}\sum_{b=1}^{W/2} s_{a,b}\cdot \log\psi_{a,b}(\boldsymbol{n}_{a,b}), \tag{13}$$

$$l_C = \left(-\sum_{a=1}^{H/2}\sum_{b=1}^{W/2}\sum_{\boldsymbol{n}\in I^4}\psi_{a,b}(\boldsymbol{n})\log_2\psi_{a,b}(\boldsymbol{n}) - C\right)^2. \tag{14}$$

(5) Update the parameters of the environment network via minimizing the cross-entropy loss l_E as

$$l_E = -z_0'\log(z_0) - z_1'\log(z_1), \tag{15}$$

where z_0 and z_1 are respectively the environment network's softmax outputs for the cover image $\boldsymbol{X}$ and the simulated stego image $\boldsymbol{Y}$, and z_0' and z_1' are their corresponding ground-truth labels.

The training iterations are set to 30000. When the training process terminates, the joint policy matrix $\boldsymbol{\Psi} = (\boldsymbol{\psi}_{a,b})^{H/2\times W/2}$ can be obtained by the policy network for an input image. The elements of the matrix can be converted into joint embedding costs according to (16). The joint embedding costs can be decomposed into additive costs by DeJoin, and then STC or SPC can be applied to message embedding.

$$\rho_{a,b}^{\boldsymbol{n}} = \ln\frac{\psi_{a,b}(\boldsymbol{n}_0)}{\psi_{a,b}(\boldsymbol{n})}, \quad \boldsymbol{n}\in I^4, \quad \boldsymbol{n}_0 = (0,0,0,0). \tag{16}$$

4 Experiments

To evaluate the effectiveness of JoPoL, extensive experiments were conducted. The experimental settings are described in Subsection 4.1. The security performance against different steganalyzers is presented in Subsection 4.2. Ablation studies on JoPoL are given in Subsection 4.3. The embedding patterns of JoPoL are analyzed in Subsection 4.4.

4.1 Settings

• **Steganographic methods.** Three steganographic methods were tested in our experiments, including a baseline state-of-the-art spatial additive method SPAR-RL-v2 [40], and two non-additive methods including [13] based on DeJoin and our proposed JoPoL. For a fair comparison, SPAR-RL-v2 was also used to obtain the additive embedding costs in both our implementation of DeJoin and the pixel-level embedding policies in JoPoL. Specifically, the parameters of JoPoL were set as follows. The batch size was set to 24. α and β in (12) were set to 1 and 10^{-7}, respectively. ξ in (10) was set to 10^7. The momentum for the moving average in the batch normalization layer was set to 0.999. The Adam optimizer with a

learning rate of 0.0001 was used for optimization. The models at the 30000-th training iteration were used to calculate the embedding costs. The steganographic methods were tested on 0.1 to 0.5 bpp (bit per pixel).

• **Steganalyzers.** Five different steganalyzers were used to evaluate the security performance of the steganographic methods, including two feature-based methods SRM [20] and maxSRMd2 [22], and three CNN-based methods Xu-Net [31], Yedroudj-Net [32], and SRNet [33].

• **Image sets.** Three image sets were used in our experiments. The ALASKA consists of 37511 images. These images were firstly demosaicked from the full resolution raw images using the executable program DCRAW, central cropped along the longer dimension as square images, and then converted to grayscale. The BOSSBase consists of 10000 images from [51]. The BOWS2 consists of 10000 images from [52]. All images from these three image sets were down-sampled to 256×256 using the "imresize" Matlab function with bicubic kernel.

In the experiments, ALASKA was used to train the cost learning models in SPAR-RL and JoPoL, while BOSSBase and BOWS2 were used to evaluate the steganographic performance. Considering that different types of steganalyzers may need different amounts of training data, their settings of image sets were different. For feature-based steganalyzers, the images from BOSSBase were randomly split into a training set and a testing set at a proportion of 1:1. For CNN-based steganalyzers, the images from BOSSBase were randomly split into a training set, a validation set and a testing set at a proportion of 4:1:5, and all images from BOWS2 were included in the training set. The security performance of the steganographic methods was evaluated as the detection error rate P_E on the testing set, which was calculated as the average of the false alarm rate and the missed detection rate.

4.2 Security performance

In this part, the security performance of different steganographic methods is compared. The experimental results are given in Table 2, and the following observations can be made.

• Compared with the baseline SPAR-RL-v2, JoPoL can significantly improve the security performance against both feature-based and CNN-based steganalyzers. For example, the improvement on 0.4 bpp is 2.40%, 2.99%, 2.96%, 1.37%, and 1.52% against SRM, maxSRMd2, Xu-Net, Yedroudj-Net, and SRNet, respectively.

• Though [13] based on DeJoin achieves satisfactory improvement over SPAR-RL-v2 against feature-based steganalyzers, it is more likely to be detected by CNN-based steganalyzers. Ref. [13] based on DeJoin is even inferior to the baseline SPAR-RL-v2 by 1.51%, 5.76%, and 2.38% against Xu-Net, Yedroudj-Net, and SRNet on 0.4 bpp. The reason is that Ref. [13] based on DeJoin introduces more embedding artifacts, which could be captured by more advanced CNN steganalyzers. Detailed analysis will be presented in Subsection 4.4.

• The two non-additive methods have comparable performance against feature-based steganalyzers, while JoPoL significantly outperforms [13] based on DeJoin against CNN-based steganalyzers. The improvement is 2.53%, 3.19%, 2.64%, 2.38%, and 2.75% against SRNet on 0.1–0.5 bpp, respectively.

4.3 Ablation studies

In this part, the impact of different factors on our proposed JoPoL is investigated, including the structure of the deep learning model, the loss function, the training strategy, and the learning algorithm. Therefore, ablation studies are conducted for different JoPoL variants. The original JoPoL is taken as a baseline.

• Variant I. S-UNIWARD is used in the pixel-level policy generation to obtain the pixel-level embedding policies, rather than SPAR-RL-v2 in the original version.

• Variant II. In the attention mechanism of the policy network, the number of attention maps and the feature maps within CNN are set to 80, rather than 12 in the original version.

• Variant III. Rather than being initialized with SPAR-RL-v2 and then fixed in the original version, the pixel-level policy generation is stochastically initialized, and then trained in an end-to-end manner.

• Variant IV. The environment assigns joint rewards based on a fixed feature critic, rather than using the gradient information from the learnable environment network in the original version. The joint reward is calculated as $s_{a,b} = -\sum_{i=1}^{4} \frac{|f^i_{a,b}(\boldsymbol{Y}) - f^i_{a,b}(\boldsymbol{X})|}{|f^i_{a,b}(\boldsymbol{X})| + \varepsilon}$, where $f^i_{a,b}(\boldsymbol{Z})$ is one of the four horizontal or vertical first-order differential residuals within the (a, b)-th 2×2 pixel block in $\boldsymbol{Z}$, and ε is a constant to avoid dividing by 0.

Table 2 P_E of steganographic methods against different steganalyzers (%)

Steganalyzer	Steganography	0.1 bpp	0.2 bpp	0.3 bpp	0.4 bpp	0.5 bpp
SRM	SPAR-RL-v2	44.94	39.09	33.96	29.51	24.87
	DeJoin	45.15	41.96	37.26	32.98	28.85
	JoPoL	45.60	40.56	35.91	31.91	27.56
maxSRMd2	SPAR-RL-v2	36.42	31.08	26.75	23.64	20.39
	DeJoin	37.31	33.14	29.35	27.04	23.49
	JoPoL	37.49	33.20	29.79	26.63	22.78
Xu-Net	SPAR-RL-v2	47.46	42.98	40.07	33.40	32.03
	DeJoin	46.28	41.21	36.69	31.89	27.11
	JoPoL	48.65	46.00	42.25	36.36	33.45
Yedroudj-Net	SPAR-RL-v2	46.26	41.82	37.07	33.33	29.30
	DeJoin	43.76	37.83	31.81	27.57	23.28
	JoPoL	47.95	43.15	38.68	34.70	31.81
SRNet	SPAR-RL-v2	35.28	27.09	20.73	16.82	12.35
	DeJoin	33.51	24.63	20.48	15.96	10.95
	JoPoL	36.04	27.82	23.12	18.34	13.70

Table 3 P_E of the baseline additive method, JoPoL, and its variants against different steganalyzers on 0.4 bpp

Steganography	SRM (%)	Xu-Net (%)
SPAR-RL-v2	29.51	33.40
JoPoL	31.91	36.36
S-UNIWARD	22.18	29.33
Varinat I	23.72	31.27
Variant II	31.97	36.97
Variant V	31.40	38.11
Variant VI	31.54	36.09
Variant VII	31.23	36.08
Variant III	31.47	36.71
Variant VIII	32.12	37.13
Variant IV	32.71	33.51
Variant IX	29.55	33.77
Variant X	30.98	36.03

• Variant V. In the attention mechanism of the policy network, 3×3 convolution is applied in the second to fourth groups, rather than 1×1 convolution in the original version.

• Variant VI. In the attention mechanism of the policy network, the number of groups are set to seven, rather than four in the original version.

• Variant VII. The original Xu-Net is applied as the environment network, rather than the enhanced Xu-Net in the original version.

• Variant VIII. The joint reward $s_{a,b}$ is calculated as the maximum of the pixel-level rewards $r_{i,j}$ within the pixel block, rather than the mean of the pixel-level rewards in the original version.

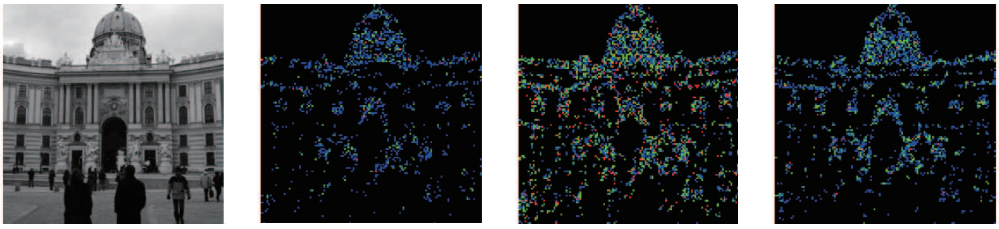
• Variant IX. Gradient descend optimizer is applied as the learning algorithm, rather than the Adam optimizer in the original version.

• Variant X. Momentum optimizer is applied as the learning algorithm, rather than the Adam optimizer in the original version.

The experimental results are shown in Table 3, and the following observations can be obtained.

• Both the original JoPoL and Variant I can significantly outperform their additive counterparts SPAR-RL-v2 and S-UNIWARD against both feature-based and CNN-based steganalyzers. The results indicate that JoPoL has great flexibility and can be combined with different additive methods.

• The methods with 12 attention maps and 80 attention maps have comparable security performance, indicating that the embedding patterns with symmetric property can share the same attention map. Besides, it can largely reduce computing resources. Specifically, the network size and computational complexity for the attention mechanism in the original JoPoL is 480 parameters and 7.86×10^6 FLOPs, while those in the Variant II is 19520 parameters and 3.20×10^8 FLOPs. Therefore, the proposed JoPoL can maintain satisfactory security performance with reduced learning complexity.



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

Figure 3 (Color online) Illustration of the cover image "01013.pgm" from BOSSBase (a), and the $\boldsymbol{N} = (\boldsymbol{n}_{a,b})^{H/2\times W/2}$ of SPAR-RL-v2 (b), Ref. [13] based on DeJoin (c), and JoPoL on 0.4 bpp (d). Each point in (b)–(d) represents an $\boldsymbol{n}_{a,b}$. For simplification, $\boldsymbol{n}_{a,b}$ where no elements are in the same direction is not given.

• Replacing the 1×1 convolution with 3×3 convolution in the attention mechanism can improve the security performance. However, deepening the attention mechanism or widening the environment network has a subtle influence on the security performance.

• JoPoL can be trained in an end-to-end learning manner. Within the proposed paradigm, the pixel-level policy generation and the attention mechanism can be effectively and jointly optimized.

• Those two joint rewards based on the gradient information, including the mean of pixel-level rewards and the maximum of the pixel-level rewards, have comparable performance. The joint reward based on the fixed feature critic can slightly improve the security performance against the feature-based steganalyzer, but largely degrade the performance against the CNN-based steganalyzer. The reason is that such a critic only considers the first-order differential feature and neglects the characteristics of other steganalyzers, which leads to poor generalization ability.

• As for the learning algorithms, the model trained with the Adam optimizer achieves the best security performance, while the one trained with the gradient descend optimizer has the worst performance. Their performance gap is significant when evaluated by either SRM or Xu-Net.

4.4 Analyzing the embedding patterns

In this subsection, the block-level embedding patterns of different methods are analyzed. Specifically, the modification map $\boldsymbol{M} = (\boldsymbol{m}_{i,j})^{H\times W}$ can be rearranged into 2×2 blocks as $\boldsymbol{N} = (\boldsymbol{n}_{a,b})^{H/2\times W/2}$, where $\boldsymbol{n}_{a,b} = (m_{i,j}, m_{i,j+1}, m_{i+1,j}, m_{i+1,j+1})$ ($i = (a-1)\times 2+1$ and $j = (b-1)\times 2+1$). The $\boldsymbol{N}$ is analyzed in three aspects as follows.

First, $\boldsymbol{N} = (\boldsymbol{n}_{a,b})^{H/2\times W/2}$ of different methods is visualized in Figure 3. Specifically, the red points, green points, and blue points respectively represent $\boldsymbol{n}_{a,b}$ with four elements, three elements, and two elements in the same direction. Generally, more red points and blue points imply a stronger effect of concentrating modification directions. It can be seen that Ref. [13] based on DeJoin obtains the strongest concentrating effect, followed by JoPoL and SPAR-RL-v2.

Second, the distribution of $\boldsymbol{n}$ is investigated. The normalized frequencies are calculated as

$$F(\boldsymbol{n}) = \frac{\sum_{a=1}^{H/2}\sum_{b=1}^{W/2}\delta(\boldsymbol{n}_{a,b}=\boldsymbol{n})}{\sum_{\tilde{\boldsymbol{n}}\in I^4}\sum_{a=1}^{H/2}\sum_{b=1}^{W/2}\delta(\boldsymbol{n}_{a,b}=\tilde{\boldsymbol{n}})}, \quad \boldsymbol{n}\in I^4, \tag{17}$$

The results over 10000 images from BOSSBase are shown in Table 4, and the following observations can be made. As for SPAR-RL-v2, no concentrating modification directions effect exists, because $F(\boldsymbol{n})$ is almost the same for those $\boldsymbol{n}$ with the same number of nonzero elements. As for [13] based on DeJoin and JoPoL, such effect is obvious. For example, $F(\boldsymbol{n})$ for $\boldsymbol{n} = (0,0,1,1)$, $(0,1,1,1)$, and $(1,1,1,1)$ are the largest under the condition of two, three, and four nonzero elements in $\boldsymbol{n}$, respectively. Furthermore, the concentrating modification directions effect for [13] based on DeJoin is stronger than that for JoPoL. In the case of four nonzero elements in $\boldsymbol{n}$, $F(+1,-1,-1,+1)$, $F(+1,+1,-1,-1)$, $F(+1,+1,+1,-1)$, and $F(+1,+1,+1,+1)$ are 0.00%, 0.02%, 0.02%, and 1.30% for [13] based on DeJoin, and 0.02%, 0.04%, 0.04%, and 0.13% for JoPoL, respectively. The stronger concentrating effect in [13] based on DeJoin implies more imbalanced distribution of $\boldsymbol{n}$, and leads to higher change rates.

Third, the relation of the distribution of $\boldsymbol{n}$ and the security performance is analyzed. According to the normalized frequencies and the change rates shown in Table 4 and the security performance in Table 2, the following inference can be made. With the guidelines of the heuristic principle, Ref. [13] based on DeJoin can concentrate modification directions and increase the security performance against

Table 4 The $F(\boldsymbol{n})$ (%) and the change rate (%) for different methods on 0.4 bpp

	SPAR-RL-v2	DeJoin	JoPoL		SPAR-RL-v2	DeJoin	JoPoL
$F(0,0,0,0)$	74.94	74.83	74.55	$F(0,+1,-1,+1)$	0.10	0.05	0.08
$F(0,0,0,+1)$	1.58	0.84	1.61	$F(0,+1,-1,-1)$	0.10	0.05	0.08
$F(0,0,0,-1)$	1.58	0.84	1.61	$F(0,-1,+1,+1)$	0.10	0.05	0.08
$F(0,0,+1,0)$	1.72	0.90	1.75	$F(0,-1,+1,-1)$	0.10	0.05	0.08
$F(0,0,-1,0)$	1.72	0.90	1.75	$F(+1,0,+1,-1)$	0.10	0.05	0.08
$F(0,+1,0,0)$	1.73	0.91	1.76	$F(+1,-1,0,-1)$	0.10	0.05	0.08
$F(0,-1,0,0)$	1.73	0.91	1.76	$F(+1,-1,+1,0)$	0.11	0.05	0.09
$F(+1,0,0,0)$	1.73	0.91	1.76	$F(-1,0,-1,+1)$	0.10	0.05	0.08
$F(-1,0,0,0)$	1.73	0.91	1.76	$F(-1,+1,0,+1)$	0.10	0.05	0.08
$F(0,0,+1,-1)$	0.31	0.02	0.12	$F(-1,+1,-1,0)$	0.11	0.05	0.09
$F(0,0,-1,+1)$	0.31	0.02	0.12	$F(+1,0,-1,-1)$	0.10	0.05	0.08
$F(0,+1,0,-1)$	0.32	0.02	0.12	$F(+1,+1,0,-1)$	0.10	0.05	0.08
$F(0,-1,0,+1)$	0.32	0.02	0.12	$F(+1,+1,-1,0)$	0.10	0.05	0.08
$F(+1,0,-1,0)$	0.32	0.02	0.12	$F(-1,0,+1,+1)$	0.10	0.05	0.08
$F(+1,-1,0,0)$	0.33	0.02	0.12	$F(-1,-1,0,+1)$	0.10	0.05	0.08
$F(-1,0,+1,0)$	0.32	0.02	0.12	$F(-1,-1,+1,0)$	0.11	0.05	0.08
$F(-1,+1,0,0)$	0.33	0.02	0.12	$F(0,+1,+1,+1)$	0.11	0.78	0.35
$F(0,0,+1,+1)$	0.31	0.81	0.47	$F(0,-1,-1,-1)$	0.10	0.79	0.35
$F(0,0,-1,-1)$	0.31	0.81	0.47	$F(+1,0,+1,+1)$	0.10	0.78	0.34
$F(0,+1,0,+1)$	0.32	0.82	0.48	$F(+1,+1,0,+1)$	0.11	0.79	0.35
$F(0,-1,0,-1)$	0.32	0.82	0.48	$F(+1,+1,+1,0)$	0.11	0.81	0.36
$F(+1,0,+1,0)$	0.32	0.84	0.49	$F(-1,0,-1,-1)$	0.10	0.78	0.35
$F(+1,+1,0,0)$	0.33	0.85	0.50	$F(-1,-1,0,-1)$	0.11	0.79	0.35
$F(-1,0,-1,0)$	0.32	0.84	0.49	$F(-1,-1,-1,0)$	0.11	0.81	0.37
$F(-1,-1,0,0)$	0.33	0.86	0.50	$F(+1,-1,-1,+1)$	0.04	0.00	0.02
$F(0,+1,-1,0)$	0.30	0.16	0.20	$F(-1,+1,+1,-1)$	0.04	0.00	0.02
$F(0,-1,+1,0)$	0.31	0.16	0.20	$F(+1,+1,-1,-1)$	0.04	0.02	0.05
$F(+1,0,0,-1)$	0.29	0.15	0.20	$F(+1,-1,+1,-1)$	0.04	0.02	0.05
$F(-1,0,0,+1)$	0.29	0.16	0.20	$F(-1,+1,-1,+1)$	0.04	0.02	0.05
$F(0,+1,+1,0)$	0.31	0.16	0.19	$F(-1,-1,+1,+1)$	0.04	0.02	0.04
$F(0,-1,-1,0)$	0.31	0.16	0.19	$F(+1,+1,+1,-1)$	0.04	0.02	0.04
$F(+1,0,0,+1)$	0.29	0.16	0.19	$F(+1,+1,-1,+1)$	0.04	0.02	0.04
$F(-1,0,0,-1)$	0.29	0.16	0.19	$F(+1,-1,+1,+1)$	0.04	0.02	0.04
$F(0,+1,+1,-1)$	0.10	0.00	0.04	$F(+1,-1,-1,-1)$	0.04	0.02	0.04
$F(0,-1,-1,+1)$	0.10	0.00	0.04	$F(-1,+1,+1,+1)$	0.04	0.02	0.04
$F(+1,0,-1,+1)$	0.10	0.00	0.04	$F(-1,+1,-1,-1)$	0.04	0.02	0.04
$F(+1,-1,0,+1)$	0.10	0.00	0.04	$F(-1,-1,+1,-1)$	0.04	0.02	0.04
$F(+1,-1,-1,0)$	0.10	0.00	0.04	$F(-1,-1,-1,+1)$	0.04	0.02	0.04
$F(-1,0,+1,-1)$	0.10	0.00	0.04	$F(+1,+1,+1,+1)$	0.04	1.30	0.13
$F(-1,+1,0,-1)$	0.10	0.00	0.04	$F(-1,-1,-1,-1)$	0.04	1.30	0.13
$F(-1,+1,+1,0)$	0.10	0.00	0.04	Change rate	10.32	14.00	10.83

feature-based steganalyzers. However, it may overly rely on few fixed embedding patterns such as $\boldsymbol{n} = (0,0,+1,+1)$, $(0,+1,+1,+1)$, and $(+1,+1,+1,+1)$. Such excessive uneven probability distribution will lead to lots of extra modifications, which may be captured by CNN-based steganalyzers. Therefore, its performance against Xu-Net, Yedroudj-Net, and SRNet is even inferior to the additive SPAR-RL-v2. On the other hand, benefiting from the data learning ability, JoPoL can differentiate the suitability of different joint embedding actions while increasing only a small number of extra modifications. Via balancing the tradeoff between concentrating modification directions and introducing additional modifications in a data-driven manner, JoPoL can improve the security performance against different kinds of steganalyzers.

5 Conclusion

In this paper, a non-additive cost learning paradigm called JoPoL is proposed, where the interactions within pixel blocks are captured and the embedding policies are defined on a block level. Inspired by

the traditional non-additive steganographic methods, the policy network is equipped with the attention mechanism, wherein the learned attention maps are utilized to evaluate the suitability of different joint embedding actions. Meanwhile, the pixel-level rewards are tuned into joint rewards to evaluate the impacts of joint embedding actions. Extensive experiments have been conducted to verify the effectiveness of JoPoL, and the following conclusions can be made.

• The architecture designed for JoPoL can learn non-additive cost functions via attention mechanism. Its end-to-end learning fashion can maximize the potential of learning abilities.

• Compared with [13] based on DeJoin which is more effective against feature-based steganalyzers but less effective against CNN-based steganalyzers, the proposed JoPoL can better resist both feature-based and CNN-based steganalyzers. The superior performance is attributed to the fact that JoPoL can concentrate modification directions with a small number of extra modifications.

The future work will focus on the following aspects. First, the case of larger pixel blocks with a low complexity will be investigated. The tradeoff between more useful correlation information and higher computation complexity should be well balanced. Second, JoPoL will be extended to the DCT domain. Utilizing neural networks to capture the complicated inter-block and intra-block correlation among DCT coefficients is a great challenge. Third, JoPoL and MCTSteg will be combined into more powerful non-additive methods. Learning non-additive steganographic costs by neural networks should have promising prospects.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 62002075, 61872244, 61872099, U19B2022), Guangdong Basic and Applied Basic Research Foundation (Grant No. 2019B151502001), and Shenzhen R&D Program (Grant No. JCYJ20200109105008228).

References

1 Shen C X, Zhang H G, Feng D G, et al. Survey of information security. Sci China Ser F-Inf Sci, 2007, 50: 273–298
2 Filler T, Judas J, Fridrich J. Minimizing additive distortion in steganography using syndrome-trellis codes. IEEE Trans Inform Forensic Secur, 2011, 6: 920–935
3 Holub V, Fridrich J. Designing steganographic distortion using directional filters. In: Proceedings of IEEE International Workshop on Information Forensics and Security, Costa Adeje, 2012. 234–239
4 Holub V, Fridrich J, Denemark T. Universal distortion function for steganography in an arbitrary domain. EURASIP J Info Secur, 2014, 2014: 1
5 Li B, Wang M, Huang J W, et al. A new cost function for spatial image steganography. In: Proceedings of IEEE International Conference on Image Processing, Paris, 2014. 4026–4210
6 Fridrich J, Kodovsky J. Multivariate Gaussian model for designing additive distortion for steganography. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, 2013. 2949–2953
7 Sedighi V, Cogranne R, Fridrich J. Content-adaptive steganography by minimizing statistical detectability. IEEE Trans Inform Forensic Secur, 2016, 11: 221–234
8 Qin X H, Li B, Huang J W. A new spatial steganographic scheme by modeling image residuals with multivariate Gaussian model. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, Brighton, 2019. 2617–2621
9 Pevny T, Filler T, Bas P. Using high-dimensional image models to perform highly undetectable steganography. In: Proceedings of International Workshop on Information Hiding, Calgary, 2010. 161–177
10 Kouider S, Chaumont M, Puech W. Adaptive steganography by oracle (ASO). In: Proceedings of IEEE International Conference on Multimedia & Expo, San Jose, 2013. 1–6
11 Li B, Wang M, Li X L, et al. A strategy of clustering modification directions in spatial image steganography. IEEE Trans Inform Forensic Secur, 2015, 10: 1905–1917
12 Denemark T, Fridrich J. Improving steganographic security by synchronizing the selection channel. In: Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security, Portland Oregon, 2015. 5–14
13 Zhang W M, Zhang Z, Zhang L L, et al. Decomposing joint distortion for adaptive steganography. IEEE Trans Circ Syst Video Technol, 2017, 27: 2274–2280
14 Li W X, Zhang W M, Chen K J, et al. Defining joint distortion for JPEG steganography. In: Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security, 2018. 5–16
15 Wang Y F, Zhang W M, Li W X, et al. Non-additive cost functions for JPEG steganography based on block boundary maintenance. IEEE Trans Inform Forensic Secur, 2021, 16: 1117–1130
16 Mo X B, Tan S Q, Li B, et al. MCTSteg: a Monte Carlo tree search-based reinforcement learning framework for universal non-additive steganography. IEEE Trans Inform Forensic Secur, 2021, 16: 4306–4320
17 Kodovsky J, Fridrich J, Holub V. Ensemble classifiers for steganalysis of digital media. IEEE Trans Inform Forensic Secur, 2012, 7: 432–444
18 Cogranne R, Sedighi V, Fridrich J, et al. Is ensemble classifier needed for steganalysis in high-dimensional feature spaces? In: Proceedings of IEEE International Workshop on Information Forensics & Security, Rome, 2015. 1–6
19 Luo X Y, Liu F L, Yang C F, et al. Modification ratio estimation for a category of adaptive steganography. Sci China Inf Sci, 2010, 53: 2472–2484
20 Fridrich J, Kodovsky J. Rich models for steganalysis of digital images. IEEE Trans Inform Forensic Secur, 2012, 7: 868–882
21 Tang W X, Li H D, Luo W Q, et al. Adaptive steganalysis against WOW embedding algorithm. In: Proceedings of ACM Workshop Information Hiding and Multimedia Security, Salzburg, 2014. 91–96
22 Denemark T, Sedighi V, Holub V, et al. Selection-channel-aware rich model for steganalysis of digital images. In: Proceedings of IEEE International Workshop on Information Forensics and Security (WIFS), Atlanta, 2014. 48–53

23 Tang W X, Li H D, Luo W Q, et al. Adaptive steganalysis based on embedding probabilities of pixels. IEEE Trans Inform Forensic Secur, 2016, 11: 734-745
24 Li B, Li Z P, Zhou S J, et al. New steganalytic features for spatial image steganography based on derivative filters and threshold LBP operator. IEEE Trans Inform Forensic Secur, 2018, 13: 1242–1257
25 Zhang H, Ping X J, Xu M K, et al. Steganalysis by subtractive pixel adjacency matrix and dimensionality reduction. Sci China Inf Sci, 2014, 57: 048101
26 Ren Y L, Zhang X P, Gu D W, et al. Efficient outsourced extraction of histogram features over encrypted images in cloud. Sci China Inf Sci, 2021, 64: 139105
27 Luo X Y, Liu F L, Yang C F, et al. Image universal steganalysis based on best wavelet packet decomposition. Sci China Inf Sci, 2010, 53: 634–647
28 Zhong K, Feng G R, Shen L Q, et al. Deep learning for steganalysis based on filter diversity selection. Sci China Inf Sci, 2018, 61: 129105
29 Tan S Q, Li B. Stacked convolutional auto-encoders for steganalysis of digital images. In: Proceedings of Signal and Information Processing Association Annual Summit and Conference (APSIPA), Siem Reap, 2014. 1–4
30 Qian Y L, Dong J, Wang W, et al. Deep learning for steganalysis via convolutional neural networks. In: Proceedings of SPIE, San Francisco, 2015
31 Xu G S, Wu H Z, Shi Y Q. Structural design of convolutional neural networks for steganalysis. IEEE Signal Process Lett, 2016, 23: 708–712
32 Yedroudj M, Comby F, Chaumont M. Yedroudj-Net: an efficient CNN for spatial steganalysis. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, Calgary, 2018. 15–20
33 Boroumand M, Chen M, Fridrich J. Deep residual network for steganalysis of digital images. IEEE Trans Inform Forensic Secur, 2019, 14: 1181–1193
34 Xu G S. Deep convolutional neural network to detect J-UNIWARD. In: Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, New York, 2017. 67–73
35 Ye J, Ni J Q, Yi Y. Deep learning hierarchical representations for image steganalysis. IEEE Trans Inform Forensic Secur, 2017, 12: 2545–2557
36 You W K, Zhang H, Zhao X F. A siamese CNN for image steganalysis. IEEE Trans Inform Forensic Secur, 2021, 16: 291–306
37 Li Q J, Feng G R, Ren Y L, et al. Embedding probability guided network for image steganalysis. IEEE Signal Process Lett, 2021, 28: 1095–1099
38 Tang W X, Tan S Q, Li B, et al. Automatic steganographic distortion learning using a generative adversarial network. IEEE Signal Process Lett, 2017, 24: 1547–1551
39 Yang J H, Ruan D Y, Huang J W, et al. An embedding cost learning framework using GAN. IEEE Trans Inform Forensic Secur, 2020, 15: 839–851
40 Tang W W, Li B, Barni M, et al. An automatic cost learning framework for image steganography using deep reinforcement learning. IEEE Trans Inform Forensic Secur, 2021, 16: 952–967
41 Tang W X, Li B, Barni M, et al. Improving cost learning for JPEG steganography by exploiting JPEG domain knowledge. IEEE Trans Circ Syst Video Technol, 2021. doi: 10.48550/arXiv.2105.03867
42 Tang W X, Li B, Tan S Q, et al. CNN-based adversarial embedding for image steganography. IEEE Trans Inform Forensic Secur, 2019, 14: 2074–2087
43 Bernard S, Bas P, Klein J, et al. Explicit optimization of min max steganographic game. IEEE Trans Inform Forensic Secur, 2021, 16: 812–823
44 Li W X, Zhang W M, Li L, et al. Designing near-optimal steganographic codes in practice based on polar codes. IEEE Trans Commun, 2020, 68: 3948–3962
45 Mnih V, Heess N, Graves A, et al. Recurrent models of visual attention. 2014. ArXiv:1406.6247
46 Xu K, Ba J, Kiros R, et al. Show, attend and tell: neural image caption generation with visual attention. In: Proceedings of International Conference on Machine Learning, 2015. 2048–2057
47 Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. In: Proceedings of International Conference on Learning Representations, 2015
48 Hu J, Shen L, Sun G. Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018. 7132–7141
49 Zhang Y L, Li K P, Li K, et al. Image super-resolution using very deep residual channel attention networks. In: Proceedings of the European Conference on Computer Vision, 2018. 286–301
50 Wang Q, Wu B G, Zhu P F, et al. ECA-Net: efficient channel attention for deep convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020. 11531–11539
51 Bas P, Filler T, Pevny T. Break our steganographic system: the ins and outs of organizing BOSS. In: Proceedings of International Workshop on Information Hiding, Prague, 2011. 59–70
52 Bas P, Furon T. Bows-2. 2007. http://bows2.ec-lille.fr