

Automating Color Image Steganography Research from First Principles

Martin Benes¹[0000–0003–0008–7524] and Rainer Böhme¹[0000–0003–4518–6227]

University of Innsbruck, Austria
`{martin.benes,rainer.boehme}@uibk.ac.at`

Abstract. This paper presents a new method for color image steganography, trained automatically using a GAN. Our minimalistic generator network for calculating the embedding costs is nearly as fast as simple analytical methods, while improving the steganographic security by up to 36% compared to the best prior art. Moreover, we completely rethink the adversarial training protocol, carefully select each component, and overcome limitations of previous designs. This offers a clean starting point for refinements to specific cover sources, and a blueprint on how to automate steganography research for other cover types.

Keywords: information hiding · color image steganography · learning-based steganography

1 Introduction

Steganography is a method of hiding secret messages in innocuous cover objects. It is a vital tool for communication under oppressive supervision, e. g., for investigative journalists, dissidents in totalitarian regimes, or undercover agents. Steganography is deemed secure if the adversary, called a steganalyst, cannot distinguish between plain covers and stego objects with an embedded message [17].

For covers with intractable distributions, such as digital images, research on steganography and steganalysis chase each other in a cat-and-mouse game [5]. In steganography by cover modification, this means adapting the cost function, i. e., the strategy of *where to change*, to the best-known steganalysis at a time [23,34]. Given these costs, coding theory offers efficient solutions for the task of *how to embed* [15]. In steganalysis, detectors are usually empirical, e. g., using statistics [6], features-based classifiers [20], or neural networks [53]. To date, designing a new steganographic method involves tedious trial-and-error loops.

In this paper, we provide a fresh start to constructing steganographic methods by combining several key ideas. We design the steganographic method automatically by leveraging *adversarial learning*, i. e., generative adversarial networks (GANs). Our generator network *predicts costs*, because they are derived from the pixel’s local neighborhood, eliminating the need for a global receptive field. This allows us to use a generator architecture that is *simple* to embed efficiently and *flexible* to embed in covers of arbitrary dimension. We select *color images* as

covers, because they are common in digital communication and yet remain underrepresented in the steganographic literature compared to grayscale [27,9]¹. Our method *models the color channels jointly*, to be able to capture the inter-channel dependencies. Failing to do so has facilitated attacks in the past [30]. Combining these ideas yields an embedding function that, when benchmarked against four tuned detectors, matches or improves upon the most secure known methods for this cover type.

This paper is structured as follows. Section 2 sets the necessary background. Section 3 summarizes the related work. Section 4 describes our method. Section 5 presents the experimental setup and the results of the evaluation. Section 6 discusses and Section 7 concludes. The source code and the trained models are available at <http://github.com/UIBK-uncover/steGAN-dalf>.

2 Background on steganography

This section sets the necessary background to steganography: the communication scenario (Sec. 2.1) and the typical design of embedding functions (Sec. 2.2).

2.1 Steganographic communication scenario

The steganographic scenario, known as the prisoners’ problem [40], involves two partners, who share a communication channel that is observed by a passive adversary. To reclaim confidentiality, the sending partner embeds a secret message $\mathbf{m}$ into an ordinarily exchanged cover object $\mathbf{x}^{(0)}$ with N elements. We assume that the message is embedded by slightly modifying the cover, $\mathbf{x}^{(0)} + \mathbf{d}$, where $\mathbf{d} \in \Delta^N$ is the change vector and Δ is the alphabet of permitted changes, e. g., $\{0, +1, -1\}$ for elements that are pixel values. The resulting stego object $\mathbf{x}^{(\mathbf{m})}$ is sent amid other (cover) objects in the communication.

The sender and recipient use a shared secret key $\mathbf{k}$. For the steganalyst who does not have $\mathbf{k}$, the changes appear as random noise. The steganalyst succeeds if she can identify that the noise present in the image carries *any* message. Shorter messages require fewer changes, which are typically harder to detect. This creates a trade-off between capacity and security that depends on the embedding rate $\alpha = \frac{|\mathbf{m}|}{N}$ bits per element (bpe). $|\mathbf{m}|$ is the message length (in bits) [17, Ch. 4].

Under Kerckhoffs’ principle [29], the designs of the embedding and extraction functions are public. The attacker is thus able to tailor detectors specifically for to the steganographic method. In addition to $\mathbf{k}$ being secret, the sender must choose a cover $\mathbf{x}^{(0)}$ that is unavailable to the steganalyst [2].

2.2 Design of the embedding function

State-of-the-art methods decompose the embedding function into calculating costs and message coding.

¹ In the last five years, 70% of works on image steganography at the specialized information hiding venues, IH&MMSec and WIFS, present empirical results for grayscale.

The **cost** $\rho_{\cdot,i}$ roughly expresses the sender’s belief about how confident the adversary’s detector is for the i -th element. Elements where a change would be obvious and easy to detect should have high costs. The costs are calculated using a cost model $\rho_{\delta,i} = f(\mathbf{x}_{\mathcal{N}(i)}^{(0)}, \delta)$ from the neighborhood $\mathbf{x}_{\mathcal{N}(i)}$ of the i -th element and with respect to the applied change $\delta \in \Delta$. Many analytically designed cost functions in the literature are symmetric, $\Delta = \{0, \pm 1\}$, and use $\rho_{0,i} = 0$. Examples include HIGH-Low-Low (HILL) [34] or Wavelet-Obtained Weights (WOW) [23]. Asymmetric cost models, $\Delta = \{0, +1, -1\}$, can prioritize one of the directions, e. g., never increment in saturated white areas. There are cases of even larger alphabets. Color-Pixel Vectors (CPV) [37] models the whole color pixel triplet jointly with 27-ary changes, $\Delta = \{000, +00, -00, 0+0, \dots, +--, --\}$. In general, cost models with larger alphabets are more flexible, at the expense of more involved coding. For example, 27-ary pixel models could, in principle, embed securely even when demosaicing traces are present, which were previously exploited to attack color image steganography [30].

The **coding** of the message $\mathbf{m}$ into the cover $\mathbf{x}^{(0)}$ involves searching for $\mathbf{d}$ that minimizes the total distortion D derived from the costs. This search is constrained by ensuring that $\mathbf{m}$ can be extracted correctly from $\mathbf{x}^{(\mathbf{m})}$ without access to $\boldsymbol{\rho}$ and $\mathbf{x}^{(0)}$. For ± 1 changes, the message is encoded to the least-significant bits of the stego object’s elements. In practice, the constrained optimization is solved by coding, e. g., with Syndrome–Trellis Codes (STCs) [16]. Like most coding methods, STCs derive the distortion as the sum of the costs of all changes made, $D = \sum_i \rho_{d_i,i}$, which implies that changes do not interact. While this assumption is unrealistic — changing a pixel may impact the costs of its neighbors — it has been established empirically that the resulting steganography has good security. Non-additive coding is not mature, hence we omit it in this paper [36].

When designing embedding functions, researchers typically skip the coding and resort to simulation [23,34,56]. The costs $\boldsymbol{\rho}$ are converted to a probability map $\beta_{\delta,i} = P(d_i = \delta \mid \mathbf{x}^{(0)}, \boldsymbol{\rho}, |\mathbf{m}|)$, where the probability is taken over the key $\mathbf{k}$ and the message $\mathbf{m}$ of fixed length. Under optimal coding, the probabilities follow a Boltzmann distribution, $\beta_{\cdot,i} = \text{softmax}(-\lambda \boldsymbol{\rho}_{\cdot,i})$. The desired message length $|\mathbf{m}|$ is controlled by the value of λ . As the relationship to the entropy $\sum_i H(\beta_{\cdot,i}) = |\mathbf{m}|$ defines λ implicitly, its value must be found numerically [14].

3 Related work

This section summarizes the prior art on color image steganography (Sec. 3.1) and GAN-based steganography (Sec. 3.2).

3.1 Steganography in color images

When the cover $\mathbf{x}^{(0)}$ is a color image, the i -th element has multiple color channels, e. g., red, green, and blue. Early works [19,18] naïvely extend grayscale steganography to color by treating the color channels as separate images (ch-by-ch), or by concatenating them into one long image (concat’d). Such strategies

assume no interaction of changes along the channel axis. Later methods, such as Clustering Modification Directions in Color (CMD-C) [45] or G-channel-related inter-channel non-additive (Gina) [47], adjust the per-channel costs post-hoc to improve the cross-channel behavior.

A notable milestone were the two ALASKA steganalysis competitions [9,10]. Not only did they establish the de-facto standard dataset, but the insights gained have reshaped the field, e. g., the surprising success of computer-vision architectures [53]. The recent literature replaces the per-channel (marginal) models by joint models of the color pixel. Color Pixel Vectors (CPV) [37] has the first analytical 27-ary cost function. Phased Training Strategy (PTS) [56] conditions the cost function on the changes applied to the other channel(s). So far, it is the only steganographic method for color that uses neural networks to (indirectly) determine the costs (see Tab. 1 below).

3.2 GAN-based steganography

Steganography is a security technique that must anticipate adversaries. This makes generative adversarial networks (GAN) [21], where two networks learn by competing against each other, a suitable choice for designing learning-based steganography. While attempts to learn the embedding function end-to-end underperform in terms of the capacity–security trade-off [41], secure methods can be learned by letting the generator network (G_ϕ) predict where to embed and the discriminator network (D_θ) learn to steganalyze. D_θ provides feedback to G_ϕ , which learns to fool D_θ . According to our overview of GAN-based steganography in Tab. 1, all existing methods train with an embedding simulator, and their G_ϕ ’s directly output embedding probabilities with explicitly constrained entropy, i. e., $\sum_i H(G_\phi(\mathbf{x}^{(0)})) = |\mathbf{m}|$. During inference, the outputs of G_ϕ are converted to costs, $\rho = \log(1 - G_\phi(\mathbf{x}^{(0)})) - \log G_\phi(\mathbf{x}^{(0)})$. Such constructions must be trained at a fixed embedding rate α , which may hinder the ability to generalize to different rates. Moreover, a fixed output entropy can only be met if the architecture of G_ϕ has a global receptive field, requiring a scalar bottleneck. Therefore, the literature employs large generators based on the U-Net architecture [39], whose repeated sub/upsampling practically constrains the cover dimensions to 256^2 .

For GAN training, it is important that the embedding simulator is differentiable to let D_θ ’s feedback flow back to G_ϕ . The literature tackles this in several ways: double-tanh reparametrization [50], proxy network [46], policy gradient, i. e., the D_θ ’s loss contains β , which lets the gradient bypass the simulation [43], or replacing the gradient with β (“straight-through”) [49]. We leave a comparison of these approaches to future work, and use Gumbel-softmax reparametrization [26], similarly to [4].

4 Design

Figure 1 shows our system model, consisting of two networks: generator (G_ϕ) and discriminator (D_θ). Their architectures are described in Sec. 4.1 and Sec. 4.2,

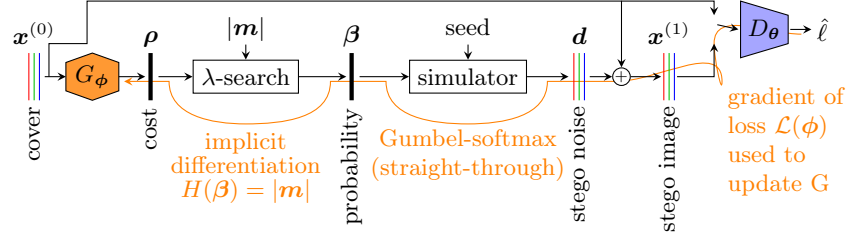


Fig. 1: Our proposed adversarial system consists of two neural networks. The generator (G_ϕ) controls where to embed by predicting costs ρ of change. The discriminator (D_θ) provides feedback to G_ϕ during training. The gradients propagate from D_θ to G_ϕ thanks to differentiable intermediate steps, adapted from [4].

respectively. The training procedure is explained in Sec. 4.3. Here we proceed with further explaining details of the system model in Fig. 1.

Forward pass For a color cover image $\mathbf{x}^{(0)}$, G_ϕ infers 27-ary costs ρ that capture all the change combinations within a color pixel triplet. In practical systems, ρ could be an input to multi-layered codes [15]. For training, ρ are converted to probabilities $\beta_{\cdot,i} = \text{softmax}(-\lambda \rho_{\cdot,i})$ with entropy $\sum_i H(\beta_{\cdot,i}) = |\mathbf{m}|$, where λ is found with binary search, preceded by an exponential boundary search. The categorical simulator is reparametrized into $d_i = \lfloor \text{softmax}(\log \beta_i + \mathbf{g}_i) \rfloor$, where $\mathbf{g} \sim \text{Gumbel}(0, 1)$. The rounding $\lfloor \cdot \rfloor$ ensures discrete stego changes. The stego noise $\mathbf{d}$ is added to $\mathbf{x}^{(0)}$ to get the stego object $\mathbf{x}^{(1)}$. The discriminator receives $\mathbf{x}^{(0)}$ or $\mathbf{x}^{(1)}$ (at a coin toss) as an input, and yields a prediction logit $\hat{\ell} \in \mathbb{R}$.

Backward pass Training of D_θ as a binary detector matches the standard deep-learning training procedure. Training of G_ϕ to fool D_θ requires that the gradients of the loss function $\mathcal{L}(\phi)$ propagate from $\hat{\ell}$ through all the steps back to G_ϕ . The non-differentiable obstacles are the simulator and the numerical λ -search. For the simulator, the Gumbel-softmax construction lets the gradients pass through the parameter β of interest by isolating it from the randomness. The rounding is ignored in the gradient calculation, which is known as the straight-through estimator (STE) of the step function [3]. As for the numerical solver, we manually rewrite the automated gradient by the gradient of the entropy constraint that is being solved. The constraint is an implicit function $h(\rho, \lambda) = 0$, which implies $\frac{\partial \rho_i}{\partial \lambda} = -\frac{\partial h}{\partial \rho_i} \left(\frac{\partial h}{\partial \lambda} \right)^{-1}$. The value of λ is taken from the forward pass.

Differences to Bernard et al. We adapt the same differentiability tricks as the adversarial method Backpack [4], and deem it relevant to highlight the differences. Backpack uses a single discriminator network D_θ rather than a GAN, and thus it is excluded from Tab. 1. Its embedding costs are acquired for a given cover by accumulating gradients passed from D_θ via the simulator. Our generator G_ϕ amortizes this effort to many covers; it carries the knowledge that Backpack must re-acquire for every embedding through repeated back-propagation.

Table 1: Prior art on GANs predicting where to embed. All methods predict probabilities and only one targets color. Various approaches have been tried to make the steganographic operation differentiable, which is essential for a GAN.

Method	Embedding domain	Emb. simulator differentiability	Generator output and dimension
Before 2020			
ASDL-GAN [46]	grayscale spatial	proxy network	probability map 1
UT-GAN [50]	grayscale spatial	double-tanh	probability map 1
JS-GAN [51,49]	grayscale JPEG	straight-through	probability map 1
After 2020			
SPAR-RL [43]	grayscale spatial	policy gradient	probability map 1
JEC-RL [44]	grayscale JPEG	policy gradient	probability map 1
UMC-GAN [55]	grayscale spatial	double-tanh (mod.)	probability map 1
AACL-GAN [25]	grayscale spatial	policy gradient	probability map 2
Steg-GMAN [24]	grayscale spatial	double-tanh	probability map 1
ARES [13]	grayscale spatial	double-tanh	probability map 1
SECA [52]	grayscale spatial	policy gradient	probability map 2
PTS [56]	color spatial	policy gradient	probability map 6
<i>Proposed</i>	color spatial	Gumbel-softmax	cost map 27

Furthermore, Backpack is designed for JPEG-compressed grayscale images and struggles to converge in the spatial domain [4, Sec. VII.E.]. Adapting Backpack to color and our approach to JPEG is beyond the scope of this paper. Thus, we defer a head-to-head comparison with Backpack to future work.

4.1 Generator

Figure 2 illustrates the architecture of our generator network. It is based on a single depthwise-separable convolution block, popularized by the Xception architecture [8]. The latent representation has 192 channels—64 channels per color channel—with a leaky rectified linear unit (leak slope 0.2, following [38]) as non-linearity. Intuitively, the depthwise convolution captures per-channel noise patterns, and the pointwise convolution models the inter-channel dependencies. There is no activation at the output of the pointwise convolution; the subsequent λ -search can be viewed as an entropy-adaptive softmax activation function.

4.2 Discriminator

Figure 3 shows our discriminator network D_{θ} , which is based on the established UcNet architecture, proposed for color steganalysis [48]. Its general idea is to gradually decrease the image resolution while increasing the number of channels. UcNet (like other steganalysis networks) keeps the original resolution for the first few layers to retain the high-frequency stego noise. The architecture

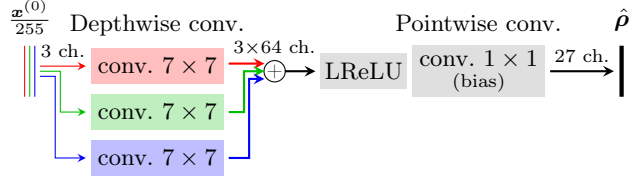


Fig. 2: The generator network (G_ϕ) is intentionally simple. It uses one depthwise-separable convolution block: each color channel is processed by 64 different learnable convolutions, concatenated along the channel axis, and passed to leaky ReLU activation functions before being aggregated to 27 cost values, one for each change direction in the color cube. G_ϕ has 15K parameters in total.

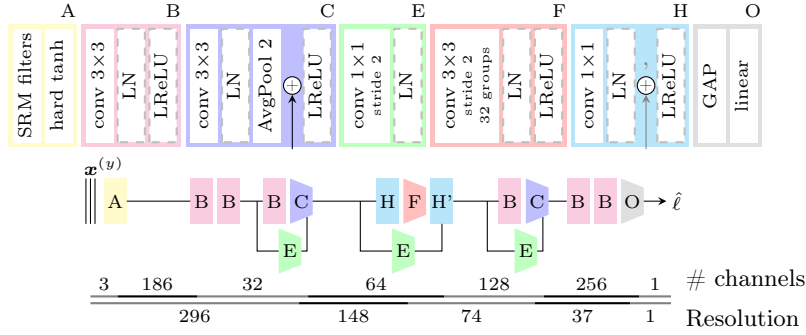


Fig. 3: The discriminator network (D_θ) is more complex to make a strong adversary. We modify UcNet [48] by replacing batch normalization with layer normalization (LN) and ReLU with leaky ReLU, as indicated by the dashed boxes. The H block doesn't use the second input, unlike H'. D_θ has 677K parameters.

of UcNet is fairly involved, with more than 10 convolutional layers and several skip connections. The first block (A) applies fixed high-pass filters, inspired by spatial rich models (SRM), followed by hard tanh activations. The final classifier (O block) contains global average pooling (GAP) and a linear layer that weighs the output features into a single output logit, $\hat{\ell} = D_\theta(\mathbf{x}^{(y)})$, where $y \in \{0, 1\}$ is the class label. We introduce two modifications that are common to GAN discriminators: (1) all batch normalization layers are replaced with layer normalization (LN) to reduce cross-sample correlations [22]; and (2) rectified linear units (ReLU) are replaced with leaky ReLU (LReLU), following [38]. A total of 677387 parameters means that our D_θ is almost 50 times larger than our G_ϕ .

4.3 Training

D_θ trains to minimize

$$\mathcal{L}(\theta) = \text{ReLU}(1 + D_\theta(\mathbf{x}^{(0)})) + \text{ReLU}(1 - D_\theta(\mathbf{x}^{(1)})). \quad (1)$$

G_ϕ minimizes an adversarial objective,

$$\mathcal{L}(\phi) = D_{\theta}(\underbrace{\mathbf{x}^{(0)} + \mathbf{d}}_{\mathbf{x}^{(1)}}) + \text{ReLU}\left(\mathbb{E}_i \underbrace{\|\text{project}(\boldsymbol{\beta}_{:,i})\|_1}_{\text{marginals}} - 0.5\right), \quad (2)$$

where $\mathbf{d} \sim \text{Categorical}(\text{softmax}(-\lambda \cdot G_\phi(\mathbf{x}^{(0)})))$ is carried out by the Gumbel-softmax reparametrization. The second term pushes the average binary change rate below the budget of 0.5 changed elements per pixel, embodying the minimum-impact paradigm [12][17, Sec. 7.4]. The projection from 27-ary probabilities to the marginals is defined as $\text{project}(\boldsymbol{\beta}_{:,i})_c = \sum_{\delta \in \Delta, \delta_c \neq 0} \beta_{\delta,i}$. Without this term, the generator is reckless and the change rate quickly reaches 100%.

The losses are intentionally asymmetric. The linear adversarial objective $\mathcal{L}(\phi)$ —often referred to as non-saturating [21]—ensures that G_ϕ receives strong, non-zero gradients even if D_θ is confident. The hinge loss $\mathcal{L}(\theta)$ bounds the gradient to a margin $[-1, 1]$, preventing training instability and gradient explosion associated with unbounded or non-saturating objectives [1].

Embedding rate for training Unlike the change probabilities $\boldsymbol{\beta}$, the steganographic costs $\boldsymbol{\rho}$ can be independent of the message length $|\mathbf{m}|$. The embedding rate α used by the generator during training influences the performance at different embedding rates. If G_ϕ is trained at low α , it identifies cheap positions, but fails to accurately localize highly detectable positions. We conjecture that a learned cost function exposed to a diverse range of α during training will develop a representation that is robust to varying message lengths during validation.

5 Experiments

5.1 Experimental setup

Dataset For the experiments, we use 80 005 color images from the ALASKA2 dataset [9]. We apply center-cropping of the original 3×512^2 down to 3×296^2 , which corresponds (in the number of values) to the conventional 512^2 of grayscale. This is based on the rationale that security scales along the number of values rather than pixels (for grayscale they are the same) [28]. The dataset was randomly split 70:15:15 into training, validation, and test sets. The high number of training images combined with our small G_ϕ effectively prevent overfitting.

Steganography We benchmark against a portfolio of relevant state-of-the-art methods. Marginal cost models are represented by HILL [34] and WOW [23], both originally designed for grayscale. We apply their conventional extensions to color, by (1) embedding channel-by-channel at equal rates [20], (2) concatenating the color channels [11], and (3) using the Gina strategy [47]. Joint cost models are represented by the analytical CPV [37] and the GAN-based PTS [56].

For PTS, we use ALASKA-trained weights, provided by the authors. As PTS networks constrain the image dimension to 3×256^2 , we match this resolution

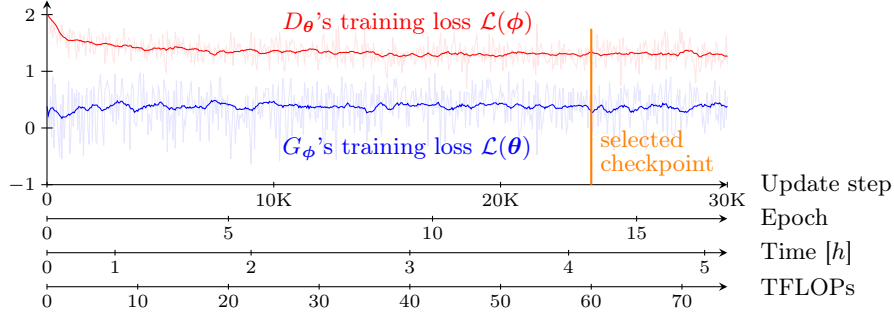


Fig. 4: The adversarial training is remarkably stable. The losses are smoothed by moving average with window size 80. FLOPs are estimated using the built-in PyTorch FLOP counter.

by top-left cropping our 3×296^2 images, giving PTS a slight advantage [28]. We assess the performance on 16K images using their test set. With such large, randomly sampled test sets, the effect of different splits is expected to be negligible.

We use common embedding rates of $\alpha \in \{1, 0.4, 0.2, 0.1\}$ bpe. For a fair comparison, the joint cost models always match the α in their marginal projections. We ensured that the simulator seed differs for each cover.

Adversarial training Here we specify our experimental settings to the general training protocol, described in Sec. 4.3. Both networks were trained on the training set for 15 epochs with minibatches of 32 covers. Each update step consists of the discriminator update, preceded by the generator update in every fifth update, similar to [1]. During the generator update, we average the 32 loss values $\mathcal{L}(\phi)$ and propagate the gradients back to G_ϕ . During the discriminator update, we recalculate β with the (just-updated) generator, average the 32 loss values $\mathcal{L}(\theta)$, and update D_θ . Both networks are trained with fixed learning rates 10^{-4} and the RMSprop optimizer. The target embedding rate α is drawn uniformly in the interval $[0.1, 1]$, and each cover uses a distinct simulator seed.

Fig. 4 shows the progress of the GAN training on an NVIDIA H100 GPU. After the initial 5K steps, when D_θ improves, G_ϕ receives sufficient traction to start learning where embedding is risky. We select the generator state after 24K steps and 4h 15 of training, where the curves had already flattened out. In a stable, converged GAN, the choice of the step should not make a big difference.

Detectors Security is measured against three kinds of deep-learning detectors: UCNNet in its vanilla form, as described in [48]; SRNet [7], a popular steganalysis-specific architecture with 6.8M parameters, adopted from grayscale by increasing the number of input channels (same as in [56]); and EfficientNet B0 [42], a vision-inspired CNN architecture with 4M parameters. For B0, we apply two techniques known to boost its performance for steganalysis: ImageNet weight pretraining [54] and removing the stem-layer stride [53]. We also evaluate against

Table 2: Computational resources used to produce the results in this paper.

Node	GPU	GPU mem. [GB]	CPU	RAM [GB]
I	NVIDIA H100	94	Intel Xeon Gold 6438Y+	1000
II	NVIDIA A40	48	AMD 7252 CPU	32
III	NVIDIA A30	24	Intel Xeon 8358 CPU	32
IV			Intel Xeon Platinum 8357	16

feature-based detectors. The spatio-color rich models (SCRM) [20] features, designed for color image steganalysis, are combined with an ensemble of Fisher’s linear discriminants (FLD). For this, we follow the training protocol in [31].

All networks are trained against each stego method, with cross-entropy loss and a curriculum along the embedding rate, $\alpha = 1 \rightarrow 0.4 \rightarrow 0.2 \rightarrow 0.1$ bpe. Each curriculum step consists of 30 epochs over the training set with the one-cycle scheduler at a peak learning rate 10^{-3} , using an Adam optimizer. The network state from the epoch with the lowest validation loss is used to measure the test-set performance. Following conventions, detection performance is reported as the probability of error, $P_E = \arg \min_{\tau} \frac{1}{2} (P_{FP}(\tau) + 1 - P_{TP}(\tau))$, where $P_{FP}(\tau)$, $P_{TP}(\tau)$ is the point on the ROC curve for the decision threshold τ .

Computational resources A single curriculum step of a detector takes about 8h on an NVIDIA A30 GPU. With one such GPU, reproducing the results from Tab. 3 in Sec. 5.2 takes 36 days. Hence, we use nodes on three clusters available to our lab. A specification of the nodes is provided in Tab. 2. The GAN training was carried out on nodes of Type I. The detectors were trained on Types I, II, and III. The embedding speed was measured on nodes of Type IV.

5.2 Results on security

Table 3 compares the detectability of the selected steganographic methods for color images across four steganalysis detectors. Against *deep learning-based detectors*—B0, SRNet, and UcNet—the proposed method demonstrates competitive security, particularly at higher embedding rates (α) and against the EfficientNet B0 architecture. For shorter messages ($\alpha = 0.1$), it is surpassed by PTS. In most configurations, it outperforms the analytical methods Gina or CPV. The comparison shifts for *feature-based detectors*, which the analytical methods are tailored to resist. In this regime, PTS offers only a marginal improvement over analytical methods, while the proposed method shows a performance lag. However, feature-based detectors are weaker than deep learning-based detectors in all conditions and therefore less relevant for practically secure steganography.

To evaluate the worst-case scenario for a steganographer, we combine the results into a single score by selecting the most powerful attacker’s detection error at each α (see “Best of above”). Under this criterion, the proposed method consistently outperforms the baseline and ranks second only to PTS at $\alpha =$

Table 3: Detector error P_E ($\uparrow$) for color image steganography. The proposed method is competitive or more secure (i. e., higher detector error) than existing methods against the best-performing detector for each embedding rate α (underlined). **Bold** marks the most secure steganographic method for the given detector and α . Results from 12K test images of the ALASKA dataset [9].

Steganalyst's detector	α	Steganographic method								
		Ch-by-ch		Concat'd		Gina		CPV	PTS	<i>Proposed</i>
		HILL	WOW	HILL	WOW	HILL	WOW			
EfficientNet B0	0.4	<u>.109</u>	.145	.150	.140	<u>.157</u>	.187	<u>.094</u>	<u>.161</u>	.229
	0.2	<u>.178</u>	.208	.226	<u>.205</u>	<u>.211</u>	.233	<u>.159</u>	<u>.232</u>	.283
	0.1	<u>.224</u>	<u>.272</u>	.289	<u>.268</u>	<u>.289</u>	<u>.295</u>	<u>.200</u>	<u>.320</u>	.331
SRNet	0.4	<u>.124</u>	<u>.136</u>	.149	<u>.147</u>	.166	<u>.170</u>	.121	.169	.244
	0.2	<u>.174</u>	<u>.175</u>	.223	<u>.204</u>	<u>.224</u>	<u>.230</u>	.189	.283	.286
	0.1	<u>.250</u>	<u>.243</u>	.280	.283	.298	.307	.262	.361	.318
UCNet	0.4	<u>.127</u>	<u>.130</u>	<u>.147</u>	<u>.147</u>	.175	.176	.215	.193	.219
	0.2	<u>.184</u>	.200	<u>.200</u>	.209	.230	.242	.283	.302	.282
	0.1	<u>.251</u>	.258	<u>.269</u>	.288	.294	.307	.335	.396	.316
SCRM features & FLD ensemble	0.4	.203	.220	.225	.224	.277	.270	.292	.296	.254
	0.2	<u>.328</u>	<u>.306</u>	<u>.323</u>	<u>.336</u>	.375	.370	.362	.361	.334
	0.1	<u>.387</u>	<u>.395</u>	<u>.396</u>	.410	.442	.438	.443	.414	.358
<u>Best of above</u>	0.4	<u>.109</u>	<u>.130</u>	<u>.147</u>	<u>.140</u>	<u>.157</u>	<u>.170</u>	<u>.094</u>	<u>.161</u>	.219
	0.2	<u>.174</u>	<u>.175</u>	<u>.200</u>	<u>.204</u>	<u>.211</u>	<u>.230</u>	<u>.159</u>	<u>.232</u>	.286
	0.1	<u>.224</u>	<u>.243</u>	<u>.269</u>	<u>.268</u>	<u>.289</u>	<u>.295</u>	<u>.200</u>	.320	.316

0.1. For $\alpha = 0.4$, we observe the largest security improvement in terms of P_E mentioned in the abstract and conclusion.

Our P_E measurements are consistent with the results reported in the literature. A comparison is provided in Tab. 5 in the appendix.

5.3 Properties of the proposed steganographic method

Next, we analyze the learnt cost function in terms of its behavior and complexity.

Fig. 5 visualizes the probability maps β for selected images; the marginal probability $[0, \frac{1}{3}]$ is quantized and mapped to a color in the particular channel. For the joint models (CPV, PTS, and the proposed), this involves projection to the marginals, as described in Sec. 4.3. The ch-by-ch and concat'd strategies appear visually similar. Gina makes spatially synchronized changes in individual channels, and touches flat regions more often. CPV is blurry and its paler hues indicate a tendency toward synchronizing across channels—gray color implies near-constant probabilities in all channels. PTS is visually distinct, and tends to modify larger regions in one channel only. The proposed method heavily synchronizes changes across color channels within a single pixel—purple regions mean

higher probabilities in red and blue, with relatively low probability in the green channel. Observe from the brick image that the proposed method departs from the other methods: it also uses the (noisy) brick faces instead of just the lining. These insights indicate that the rules established for constructing analytical costs [33, Sec. III] may not be the only way to achieve secure steganography.

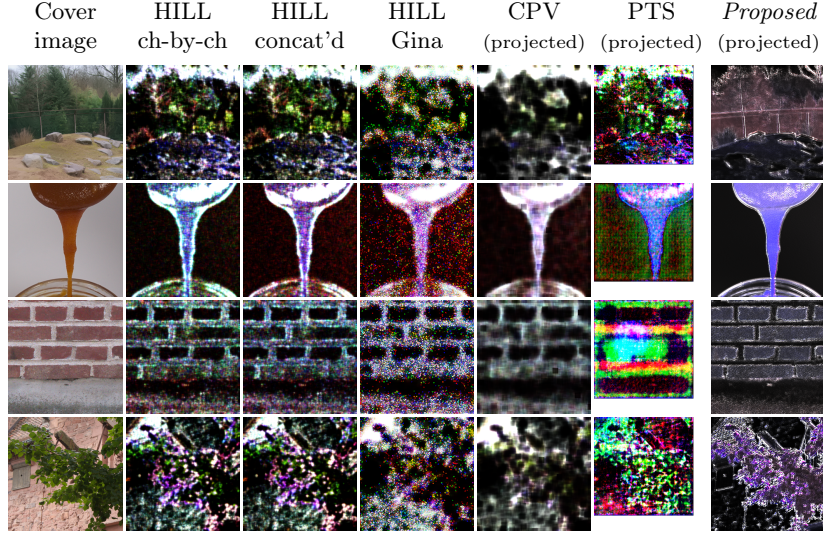
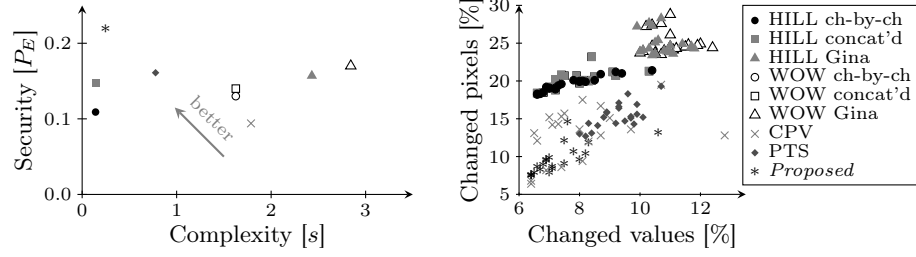


Fig. 5: Where different methods embed: cover image (leftmost) and maps of the binary probability of change for selected methods. (See appendix for WOW.) Dark pixels are changed less likely than bright ones. Red tones mean that the red channel is changed with higher probability than green or blue. The greyish tone of the proposed method means that it tends to change pixels synchronously, which may explain its superior security especially for higher embedding rates.

We further inspect the behavior in Fig. 6. The left facet (Fig. 6a) puts the security in relation to the complexity of the method. The complexity is measured as the time to calculate the cost function, averaged over 32 randomly drawn images from the test set. For the security, we take the P_E of the best detector at $\alpha = 0.4$. Our cost function is more secure than all other methods, while maintaining a low complexity, comparable to the analytical cost functions.

The right facet (Fig. 6b) quantifies the synchronization behavior of the compared methods at $\alpha = 0.4$ bpe. The joint models of CPV and the proposed method change fewer pixels, followed by PTS. Gina leaves the largest share of changed pixels, as well as values. Comparing this with the detectability in Tab. 3, synchronizing along the channel axis does not seem to be universally redemptive. A very conservative CPV strategy strongly synchronizes the changes, and yet



(a) Average time to calculate the cost ($\downarrow$) for the resolution 512^2 vs. the best detector's error, P_E ($\uparrow$), at $\alpha = 0.4$ bpe. The proposed method is in a sweet spot. (b) Different method's tendency to synchronize embedding changes to the triplets of the same pixel. Scatterplot of 30 random images from the test set.

Fig. 6: Differences in performance and behavior in color of the tested steganographic methods. The proposed method improves security while keeping complexity low. Its behavior is consistent with methods that use joint pixel models.

fails under steganalysis with EfficientNet B0. This calls into question whether the minimum-impact paradigm still applies under learning-based steganalysis.

Figure 7 reports the total embedding time on a CPU for the benchmarked methods. All methods use the same simulator, proposed in [14]. The HILL cost function is extremely cheap and, hence, it is the fastest. Our method ranks just after. Its slower embedding simulation is because the objective function, called repeatedly in the λ -search, must do more work for 27-ary costs. The cost calculation of PTS is unexpectedly fast, given that it involves inference of three large networks. Cost calculation in Gina takes longer, because it involves sorting and calculating quantiles. The last is CPV, in particular due to the slow simulator. We acknowledge a potential bias towards newer methods, because their implementations may have had access to more efficient programming frameworks. To account for potential performance, we refer the reader to the back-of-the-envelope complexity calculation (Tab. 4) in the appendix. Overall, PTS, the closest contender, has an efficient implementation and will unlikely beat the performance of the proposed method.

6 Discussion

Our training protocol represents a complete redesign compared to all existing GAN-based methods. We enable a flexible generator architecture, arbitrary cover dimensions, and less restricted training conditions, e. g., varying embedding rates for training.

Our results demonstrate that ad-hoc extensions of grayscale designs to color do not generalize well to unanticipated detectors. Adversarially learnt methods may finally be the tool that helps in finding more robust solutions. In this sense, the proposed method means a fresh start for color image steganography.

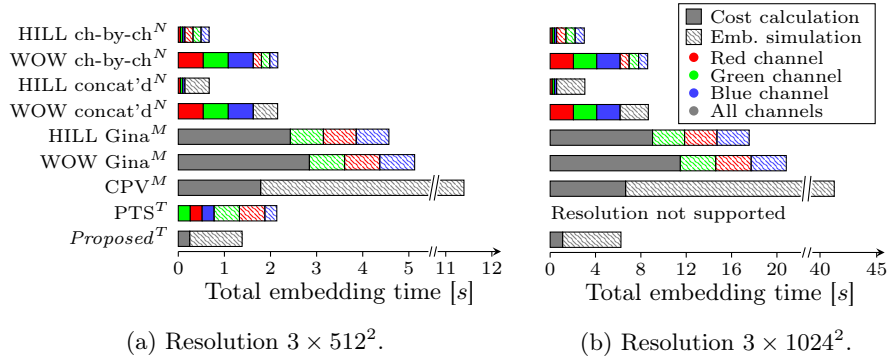


Fig. 7: Breakdown of embedding time for the color image methods in this study. Averages over 32 images on a 32-core Intel Xeon Platinum 8358 CPU with 2.6 GHz and 16GB RAM. Where available, we use the authors’ implementation in Matlab (^{M}) and PyTorch (^{T}), otherwise we reimplement in NumPy (^{N}). Only HILL is faster than the proposed method. Note the different x -axes.

Using adversarial learning in steganography research has several drawbacks. First, automating the design of steganographic functions comes with substantially higher computational overhead due to training. This could increase the cost of steganographic research, especially if tuning steganographic methods to the cover source becomes a convention. Second, the GAN training may be unstable in some cases. We observed failing convergence, for instance, with momentum-based optimizers, U-Net-based generator, B0-based discriminator, and more. Finally, employing neural networks may reduce the transparency of steganographic designs. However, our work replaces tediously crafted heuristics with a network so simple that its weights could, in principle, be manually interpreted.

Although our approach has been demonstrated using uncompressed color images, it is likely not limited to this domain. Future work could adapt similar ideas, evolving better cost functions for grayscale images, JPEG-compressed images, and other cover types, such as compressed video [32] or network traffic [35].

7 Conclusion

Automating information hiding research with learning on both sides seems inevitable, but it requires care and precision. This statement is evidenced by a number of dead ends of our own research, as well as the limited potential of the discussed related work. Our new training protocol for GAN-based steganography demonstrates that returning to first principles is rewarded with superior solutions.

Our proposed method for color image steganography improves security by up to 36% compared to state-of-the-art methods, in spite of its simpler and more flexible construction. Its high efficiency demonstrates that security improvements do not have to come at the cost of higher complexity.

Acknowledgments. We thank Tomáš Pevný for his help with differentiable steganography, Benedikt Lorch for his suggestion to make the generator model simple, and Nora Hofer, Simon Koch, Verena Lachner, and Kamila Nykiel for their feedback and helpful comments. The computational results have been achieved using the Vienna Scientific Cluster (VSC), and the LEO HPC infrastructure of the University of Innsbruck.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: Precup, D., Teh, Y.W. (eds.) *International Conference on Machine Learning (ICML)*. pp. 214–223. PMLR, PMLR (2017)
2. Beneš, M., Böhme, R.: Kerckhoffs in prison: A study of the steganalyst’s knowledge. In: *International Workshop on Information Forensics and Security (WIFS)*. pp. 1–6. IEEE (2024)
3. Bengio, Y., Léonard, N., Courville, A.: Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR* **abs/1308.3432** (2013)
4. Bernard, S., Bas, P., Klein, J., Pevný, T.: Backpack: A backpropagable adversarial embedding scheme. *IEEE Transactions of Information Forensics and Security (TIFS)* **17**, 3539–3554 (2022)
5. Böhme, R.: An epistemological approach to steganography. In: Katzenbeisser, S., Sadeghi, A.R. (eds.) *Information Hiding (IH)*. pp. 15–30. LNCS, Springer, Berlin Heidelberg (2009)
6. Böhme, R., Westfeld, A.: Breaking Cauchy model-based JPEG steganography with first order statistics. In: Samarati, P., Ryan, P., Gollmann, D., Molva, R. (eds.) *European Symposium on Research Computer Security (ESORICS)*. pp. 125–140. LNCS, Springer, Berlin Heidelberg (2004)
7. Boroumand, M., Chen, M., Fridrich, J.: Deep residual network for steganalysis of digital images. *IEEE Transactions of Information Forensics and Security (TIFS)* **14**(5), 1181–1193 (2019)
8. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 1800–1807. IEEE Computer Society (2017)
9. Cogranne, R., Giboulot, Q., Bas, P.: The ALASKA steganalysis challenge: A first step towards steganalysis. In: Cogranne, R., Verdoliva, L., Lyu, S., Troncoso-Pastoriza, J., Zhang, X. (eds.) *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. pp. 125–137. ACM (2019)
10. Cogranne, R., Giboulot, Q., Bas, P.: ALASKA#2: Challenging academic research on steganalysis with realistic images. In: *International Workshop on Information Forensics and Security (WIFS)*. pp. 1–5. IEEE (2020)
11. Cogranne, R., Giboulot, Q., Bas, P.: Steganography by minimizing statistical detectability: The cases of JPEG and color images. In: Riess, C., Schirmacher, F., Amerini, I., Bestagini, P., Pevný, T. (eds.) *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. pp. 161–167. ACM (2020)
12. Crandall, R.: Some notes on steganography. Posted on steganography mailing list **1998**(1), 6 (1998)

13. Cui, Q., Zhou, Z., Meng, R., Wang, S., Yan, H., Wu, J.: ARES: On adversarial robustness enhancement for image steganographic cost learning. *IEEE Transactions on Multimedia (TMM)* **26**, 6542–6553 (2024)
14. Filler, T., Fridrich, J.: Gibbs construction in steganography. *IEEE Transactions of Information Forensics and Security (TIFS)* **5**(4), 705–720 (2010)
15. Filler, T., Fridrich, J.: Minimizing additive distortion functions with non-binary embedding operation in steganography. In: *International Workshop on Information Forensics and Security (WIFS)*. pp. 1–6. IEEE (2010)
16. Filler, T., Judas, J., Fridrich, J.: Minimizing additive distortion in steganography using syndrome–trellis codes. *IEEE Transactions of Information Forensics and Security (TIFS)* **6**(3-2), 920–935 (2011)
17. Fridrich, J.: *Steganography in digital media: Principles, algorithms, and applications*. Cambridge University Press (2009)
18. Fridrich, J., Goljan, M., Du, R.: Detecting LSB steganography in color and grayscale images. *Multimedia* **8**(4), 22–28 (2001)
19. Fridrich, J., Long, M.: Steganalysis of LSB encoding in color images. In: *International Conference on Multimedia and Expo (ICME)*. vol. 3, pp. 1279–1282. IEEE (2000)
20. Goljan, M., Fridrich, J., Cogramne, R.: Rich model for steganalysis of color images. In: *International Workshop on Information Forensics and Security (WIFS)*. pp. 185–190. IEEE (2014)
21. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. *Advances in Neural Information Processing Systems (NIPS)* **27**, 2672–2680 (2014)
22. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.: Improved training of Wasserstein GANs. vol. 30, pp. 5767–5777 (2017)
23. Holub, V., Fridrich, J.: Designing steganographic distortion using directional filters. In: *International Workshop on Information Forensics and Security (WIFS)*. pp. 234–239. IEEE (2012)
24. Huang, D., Luo, W., Liu, M., Tang, W., Huang, J.: Steganography embedding cost learning with generative multi-adversarial network. *IEEE Transactions of Information Forensics and Security (TIFS)* **19**, 15–29 (2024)
25. Huang, D., Luo, W., Zheng, P., Huang, J.: Automatic asymmetric embedding cost learning via generative adversarial networks. In: *International Conference on Multimedia (MM)*. pp. 8316–8326. ACM (2023)
26. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with Gumbel-softmax. In: *International Conference on Learning Representations (ICLR)* (2017)
27. Ker, A., Bas, P., Böhme, R., Cogramne, R., Craver, S., Filler, T., Fridrich, J., Pevný, T.: Moving steganography and steganalysis from the laboratory into the real world. In: Puech, W., Chaumont, M., Dittmann, J., Campisi, P. (eds.) *Information Hiding and Multimedia Security (IH&MMSec)*. pp. 45–58. ACM (2013)
28. Ker, A., Pevný, T., Kodovský, J., Fridrich, J.: The square root law of steganographic capacity. In: Ker, A., Dittmann, J., Fridrich, J. (eds.) *Multimedia & Security (MMSec)*. pp. 107–116. ACM (2008)
29. Kerckhoffs, A.: La cryptographie militaire. *Journal des Sciences Militaires* **9**, 5–38 (1883)
30. Kirchner, M., Böhme, R.: "Steganalysis in Technicolor": Boosting WS detection of stego images from CFA-interpolated covers. In: *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. pp. 3982–3986. IEEE (2014)

31. Kodovský, J., Fridrich, J., Holub, V.: Ensemble classifiers for steganalysis of digital media. *IEEE Transactions of Information Forensics and Security (TIFS)* **7**(2), 432–444 (2012)
32. Lachner, V., Böhme, R.: A critical survey on video steganography: Trends, pitfalls, and recommendations. *ACM Computing Surveys (CSUR)* **58**(281), 1–35 (2026)
33. Li, B., Tan, S., Wang, M., Huang, J.: Investigation on cost assignment in spatial image steganography. *IEEE Transactions of Information Forensics and Security (TIFS)* **9**(8), 1264–1277 (2014)
34. Li, B., Wang, M., Huang, J., Li, X.: A new cost function for spatial image steganography. In: *International Conference on Image Processing (ICIP)*. pp. 4206–4210. IEEE (2014)
35. Lubacz, J., Mazurczyk, W., Szczypiorski, K.: Principles and overview of network steganography. *IEEE Communications Magazine* **52**(5), 225–229 (2014)
36. Pevný, T., Ker, A.: Exploring non-additive distortion in steganography. In: Böhme, R., Pasquini, C., Boato, G., Schöttle, P. (eds.) *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. pp. 109–114. ACM (2018)
37. Qin, X., Li, B., Tan, S., Zeng, J.: A novel steganography for spatial color images based on pixel vector cost. *IEEE Access* **7**, 8834–8846 (2019)
38. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. In: Bengio, Y., LeCun, Y. (eds.) *International Conference on Learning Representations (ICLR)* (2016)
39. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds.) *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. pp. 234–241. LNCS, Springer, Berlin Heidelberg (2015)
40. Simmons, G.: The prisoners’ problem and the subliminal channel. In: *Advances in Cryptology (CRYPTO)*. pp. 51–67. Plenum Press, New York (1984)
41. Tan, J., Liao, X., Liu, J., Cao, Y., Jiang, H.: Channel attention image steganography with generative adversarial networks. *IEEE Transactions on Network Science and Engineering (TNSE)* **9**(2), 888–903 (2022)
42. Tan, M., Le, Q.: EfficientNet: Rethinking model scaling for convolutional neural networks. In: Chaudhuri, K., Salakhutdinov, R. (eds.) *International Conference on Machine Learning (ICML)*. pp. 6105–6114. PMLR, PMLR (2019)
43. Tang, W., Li, B., Barni, M., Li, J., Huang, J.: An automatic cost learning framework for image steganography using deep reinforcement learning. *IEEE Transactions of Information Forensics and Security (TIFS)* **16**, 952–967 (2021)
44. Tang, W., Li, B., Barni, M., Li, J., Huang, J.: Improving cost learning for JPEG steganography by exploiting JPEG domain knowledge. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)* **32**(6), 4081–4095 (2022)
45. Tang, W., Li, B., Luo, W., Huang, J.: Clustering steganographic modification directions for color components. *IEEE Signal Processing Letters (SPL)* **23**(2), 197–201 (2016)
46. Tang, W., Tan, S., Li, B., Huang, J.: Automatic steganographic distortion learning using a generative adversarial network. *IEEE Signal Processing Letters (SPL)* **24**(10), 1547–1551 (2017)
47. Wang, Y., Zhang, W., Li, W., Yu, X., Yu, N.: Non-additive cost functions for color image steganography based on inter-channel correlations and differences. *IEEE Transactions of Information Forensics and Security (TIFS)* **15**, 2081–2095 (2020)
48. Wei, K., Luo, W., Tan, S., Huang, J.: Universal deep network for steganalysis of color image based on channel representation. *IEEE Transactions of Information Forensics and Security (TIFS)* **17**, 3022–3036 (2022)

49. Yang, J., Liao, Y., Shang, F., Kang, X., Chen, Y., Shi, Y.Q.: JPEG image steganography with automatic embedding cost learning. *International Journal on Intelligent Systems* **2025**(1), 5309734 (2025)
50. Yang, J., Ruan, D., Huang, J., Kang, X., Shi, Y.Q.: An embedding cost learning framework using GAN. *IEEE Transactions of Information Forensics and Security (TIFS)* **15**, 839–851 (2020)
51. Yang, J., Ruan, D., Kang, X., Shi, Y.Q.: Towards automatic embedding cost learning for JPEG steganography. In: Cогranne, R., Verdoliva, L., Lyu, S., Troncoso-Pastoriza, J., Zhang, X. (eds.) *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. pp. 37–46. ACM (2019)
52. Ye, M., Zhou, S., Luo, W., Tan, S., Huang, J.: GAN-based symmetric embedding costs adjustment for enhancing image steganographic security. In: Cai, J., Kankanhalli, M., Prabhakaran, B., Boll, S., Subramanian, R., Zheng, L., Singh, V., César, P., Xie, L., Xu, D. (eds.) *International Conference on Multimedia (MM)*. pp. 7046–7054. ACM (2024)
53. Yousfi, Y., Butora, J., Fridrich, J., Fuji Tsang, C.: Improving EfficientNet for JPEG steganalysis. In: Borghys, D., Bas, P., Verdoliva, L., Pevný, T., Li, B., Newman, J. (eds.) *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. pp. 149–157. ACM (2021)
54. Yousfi, Y., Butora, J., Khvedchenya, E., Fridrich, J.: ImageNet pre-trained CNNs for JPEG steganalysis. In: *International Workshop on Information Forensics and Security (WIFS)*. pp. 1–6. IEEE (2020)
55. Zhao, J., Wang, S.: A stable GAN for image steganography with multi-order feature fusion. *Neural Computing and Applications* **34**(18), 16073–16088 (2022)
56. Zhou, S., Ye, M., Luo, W., Liao, X., Wei, K.: Color image steganography using generative adversarial networks with a phased training strategy. In: Agarwal, S., Craver, S., Jia, S., Wong, C.W., Tondi, B. (eds.) *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. pp. 142–152. ACM (2025)

Appendix

Back-of-envelope complexity calculation

HILL The HILL cost from [34, Eq. 8] applied to a grayscale image $\mathbf{x}$ with N pixels has a complexity of

$$\underbrace{(3 + 2 + 3 + 2)N}_{\text{KB (sep.)}} + \underbrace{1N}_{|\cdot|} + \underbrace{(3 + 2 + 3 + 2)N}_{\text{AVG3 (sep.)}} + \underbrace{1N}_{\frac{1}{\cdot}} + \underbrace{(15 + 14 + 15 + 14)N}_{\text{AVG15 (sep.)}} = 84N.$$

WOW The WOW cost, specified in [23, Sec. 3.1-2] for a single channel as $\sum_{\mathbf{k} \in \mathcal{K}_{\text{Dau16}}} (\max(|\mathbf{x} * \mathbf{k}| * |\mathbf{k}^{(180^\circ)}|, 10^{-16}))^{-1}$, has a complexity

$$\underbrace{3}_{|\mathcal{K}|} \cdot \underbrace{[(16 + 15 + 16 + 15)N]_{* \mathbf{k}}}_{\text{}} + \underbrace{1N}_{|\cdot|} + \underbrace{(16 + 15 + 16 + 15)N}_{* |\mathbf{k}^{(180^\circ)}|} + \underbrace{1N}_{\frac{1}{\cdot}} + \underbrace{1N}_{\max} + \underbrace{2N}_{\Sigma} = 383N$$

CPV The CPV cost described in [37, Sec. III] for color has a complexity of

$$\underbrace{3(25 + 24 + 9 + 8)N}_{\text{high-pass}} + \underbrace{19 \cdot 8N}_{\text{correlation}} \underbrace{(27 \cdot 3 + 1)}_{\text{sensitivity}} + \underbrace{27(15 + 14 + 15 + 14)N}_{\text{low-pass}} = 14\,228N$$

Table 4: Best-effort estimates of FLOPs needed to calculate the color costs for a cover image with N pixels. Gina is the only non-linear method, due to sorting. WOW and CPV could benefit from more efficient implementations.

	Steganographic method					
	HILL	WOW	Gina	CPV	PTS	<i>Proposed</i>
FLOPS	$252N$	$1\,149N$	$\mathcal{O}(\text{cost}) + (54 + \log N)N$	$14\,228N$	$322\,162N$	$28\,992N$
Slowdown (vs. HILL)	$1\times$	$4.6\times$	depends on N	$56\times$	$1\,280\times$	$115\times$

Gina Gina cost adjustments, described in [47, Sec. V], have a complexity of

$$2 \cdot 4 \cdot \underbrace{\frac{15N}{4}}_{\text{R,B}} + 4 \cdot \underbrace{\frac{37}{12}N}_{\text{G}} + 3 \cdot 4 \cdot \underbrace{\frac{4}{4}N}_{\div 9} \approx 54N.$$

Provided a marginal cost function of complexity $\mathcal{O}(\text{cost})$, the total complexity of Gina (excluding the CMD-C simulator) is

$$\mathcal{O}(\text{cost}) + \underbrace{N \log N}_{\text{sort}} + 54N$$

PTS cost The PTS cost is calculated per-channel using three dual-branch networks of the U-Net architecture, shown in [56, Fig. 2]. The Conv block of i input channels, o output channels and input resolution N has complexity

$$\underbrace{(9+8)i^2N}_{\text{conv 1}} + \underbrace{6iN}_{\text{BN+LReLU}} + \underbrace{(9+8)io\frac{N}{4}}_{\text{conv 2}} + \underbrace{6o\frac{N}{4}}_{\text{BN+LReLU}} = [17i^2 + 6i + 4.25io + 1.5o]N.$$

The Deconv block has complexity

$$\underbrace{(25+24)i\frac{i}{2}N}_{\text{conv 1}} + \underbrace{6\frac{i}{2}N}_{\text{BN+LReLU}} + \underbrace{(25+24)\frac{i}{2}o4N}_{\text{conv 2}} + \underbrace{6o4N}_{\text{BN+LReLU}} = [24.5i^2 + 3i + 98io + 24o]N.$$

Considering all the layers and all three networks, the PTS method has complexity of $322\,160N$ FLOPs.

Proposed cost The complexity of the proposed cost is

$$\underbrace{3 \cdot 64 \cdot (49 + 48)N}_{\text{depthwise}} + \underbrace{27 \cdot (192 + 191 + 1)N}_{\text{pointwise}} = 28\,992N.$$

Table 5: Validation that our P_E measurements are in line with the literature. We get the same orders of magnitude. The difference in [56] against neural network detectors is likely because our curriculum learns stronger detectors. [20,37] use the more homogeneous BOSSBase, while we use the ALASKA dataset.

Ref.	Condition	Theirs	Ours	Comment
[56]	PTS (0.4) vs. SCRM	.308	.296	
	PTS (0.2) vs. SCRM	.411	.414	
	PTS (0.4) vs. SRNet	.402	.283	curriculum
	PTS (0.2) vs. SRNet	.461	.361	curriculum
	PTS (0.4) vs. UCNNet	.236	.193	curriculum
	PTS (0.2) vs. UCNNet	.363	.302	curriculum
	Gina HILL (0.4) vs. SCRM	.304	.277	
	Gina HILL (0.2) vs. SCRM	.394	.375	
	Gina HILL (0.4) vs. SRNet	.300	.166	curriculum
	Gina HILL (0.2) vs. SRNet	.337	.224	curriculum
	Gina HILL (0.4) vs. UCNNet	.185	.175	
	Gina HILL (0.2) vs. UCNNet	.265	.230	curriculum
[37]	CPV (0.4) vs. SCRM	.213	.291	different dataset
	CPV (0.2) vs. SCRM	.346	.362	— ” —
	CPV (0.1) vs. SCRM	.432	.443	— ” —
[20]	WOW ch-by-ch (0.4) vs. SCRM	$\sim .180$	.220	different dataset
	WOW ch-by-ch (0.2) vs. SCRM	$\sim .300$	.306	— ” —
	WOW ch-by-ch (0.1) vs. SCRM	$\sim .400$	.395	— ” —

Legend: $\sim$ means that the value is extracted from a plot.

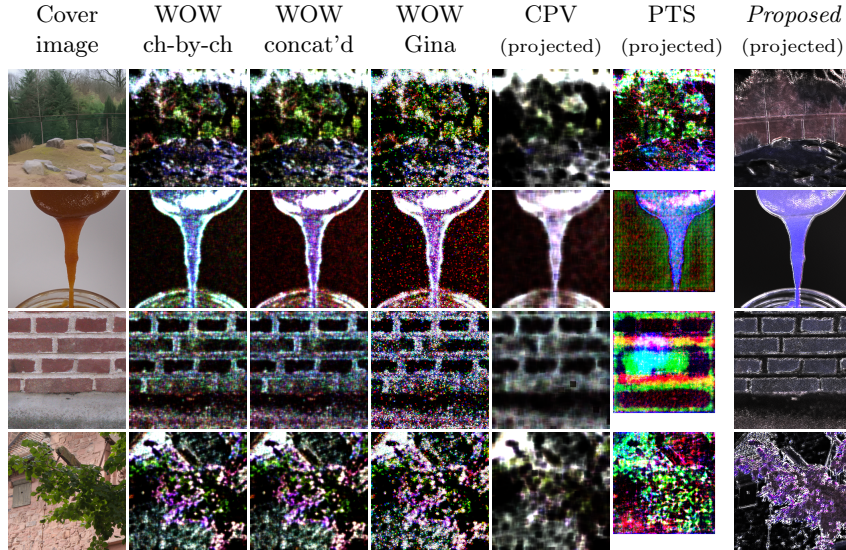


Fig. 8: Version of Fig. 5 with WOW stego method.