SvedbergOpen
DISSEMINATION OF KNOWLEDGE
International Journal of Artificial Intelligence and Machine Learning
Publisher's Home Page: https://www.svedbergopen.com/
Research Paper Open Access

Istego100k++: A Large-Scale GAN-Based Steganalysis Dataset Using Adversarially Learned Embedding Costs

Saif Salah Al-Din Affat[1*], Ismail Taha Ahmed[2]

[1,2] College of Computer Science and Information Technology University of Anbar, Anbar, Iraq
* Corresponding author's Email:saifsalah375@gmail.com
Research supervisor's email : ismail.taha@uoanbar.edu.iq

Abstract— Steganalysis benchmarks built with fixed distortion functions — such as nsF5, J-UNIWARD, and UERD — are increasingly inadequate against GAN-based steganography, where adversarially learned embedding costs bypass traditional rich-model detectors. To address this gap, we introduce IStego100K++, a large-scale GAN-generated JPEG stego dataset of 100,071 cover–stego image pairs at 1024×1024 resolution. Stego images were produced using a pre-trained UT-GAN generator with a U-Net architecture and a differentiable Double-Tanh embedding simulator, covering payload rates of {0.1–0.5} bpp and JPEG quality factors of {75–95}, achieving a mean PSNR of 74.52 dB, a mean SSIM of 0.9994, and 37.9% of images recording zero measurable distortion after JPEG recompression. Evaluated against two established rich-model steganalyzers — DCTR and GFR — detection accuracy drops from 79.55% on the original IStego100K to approximately 50% across all tested payload rates and quality conditions. Additional evaluation with CNN-based steganalyzers (SRNet-KV and XuNet) yields AUC values of 0.6953 and 0.6901 respectively, confirming that while CNN detectors outperform rich-model methods, IStego100K++ remains a significantly more challenging benchmark for next-generation universal steganalysis research.

Keywords— Image Steganography; Steganalysis; Generative Adversarial Network; UT-GAN; CNN; JPEG Compression.

1. INTRODUCTION

Steganography is a field that studies how to send information safely by hiding it in many types of digital media, like photos, text, audio, and video files [1,2]. Because information transmission technology is changing so quickly and the internet is being used so much to send and receive data, this field has become increasingly vital[3,2]. It is common to hide a message "in plain sight" in a medium called a cover or stego cover[4]. The technique involves changing the cover work in a way that is not noticeable, so that the original and modified data look the same[4]. Figure 1 shows how steganography can hide private information without making it look like it is encrypted. Steganography's main purpose is to prevent unauthorized access to data[3]. It is a critical technique for making sure that information is sent safely and with good quality[2].

Fig.1. General architecture of steganography[5].

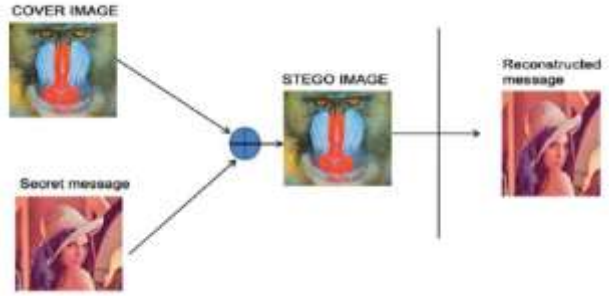


Image steganography includes several approaches ranging from basic spatial domain alteration such as least significant bit (LSB) to advanced transform domain methods using deep learning. Spatial methods have a great deal of capacity, but are less stable than transform domain approaches. The most effective systems depend on adaptive and machine learning models to identify the best trade-offs between security, capacity, and invisibility[6-7,8]. The present works on image steganography offer many solutions for data hiding. These techniques often present important drawbacks and hence they are not suitable for robust and secure applications**.**

Many modern techniques face a compromise among payload capacity, security, and the visual fidelity of the stego-image[9]. Many existing methods, however, suffer from poor embedding efficiency, poor image quality, and are too hard to compute[10]. Most early strategies are based on conventional, fixed algorithms that are easily detectable by adversaries[11]. Popular methods, such as least significant bit (LSB) replacement, are vulnerable to statistical attacks, such as regular singular (RS) analysis, despite their simplicity[11]. This indicates the high requirement of more modern techniques, which may provide security, capacity and stealth simultaneously.

Steganography methods can be classified by the domain where the information is concealed. Methods in the Spatial Domain— These techniques (LSB, PVD, EMD, PIT) embed information by directly modifying the pixel values of the cover picture[10]. Frequency (Transform) Techniques Domain: These approaches mask information by changing coefficients in a transform domain of the image such as the Discrete Cosine Transform (DCT)[11].

Steganography is driven by the need for safe and hidden communication in a world where everything is connected online. However, the limitations of conventional approaches, including poor payload capacity, being easy to find, and not being very strong, necessitate the development of more robust and secure methods to deal with new security threats[12].

To overcome these limitations, this paper introduces IStego100K++, a large-scale GAN-generated stego dataset that leverages adversarially learned embedding costs to produce stego images that are statistically undetectable by conventional rich-model steganalyzers, thereby providing a more challenging and realistic benchmark for next-generation steganalysis research.

The remainder of this paper is organized as follows: Section 2 presents the related works. The proposed method is described in Section 3. Section 4 discusses the experimental results. Finally, Section 5 concludes the paper.

2. RELATED WORKS

Over the last twenty years, research on image steganography and steganalysis has advanced significantly. There are two main types of existing approaches: traditional handcrafted methods and methods based on deep learning. This section reviews the most relevant works in each area and points out the problems that led to the suggested framework.

2.1. Traditional Steganography and Steganalysis Techniques

The basic idea behind early adaptive steganography algorithms is to reduce embedding distortion as much as possible. [13] proposed HUGO in the high dimensional feature space of pixel differences. HUGO modifies textured areas to render them less statistically detectable. HUGO does not perform well against strong detectors, as it only relies on hand-engineered heuristics. It also has less security against rich-model steganalyzers. Holub and Fridrich [14] came up with WOW (Wavelet Obtained Weights), which uses directional high-pass filters to create distortion and gives low cost to areas with complex textures. Later, the same authors proposed S-UNIWARD [15], a universal distortion function that may be used in spatial, JPEG, and side-informed domain s-UNIWARD is secure due to measuring distortion in wavelet domain, but the computation of cost map is resource consuming and its performance degrades against advanced CNN-based detectors. Fridrich and Kodovsky [16] created the Spatial Rich Model (SRM) for steganalysis by extracting a high dimensional feature vector from a large set of high-pass filter residuals. SRM is the gold standard for hand-made steganalysis and is typically used with the ensemble classifier of Kodovský et al. Although SRM works effectively, it demands careful feature engineering and its fixed filter bank is not adaptable to new steganographic systems. Filler et al. [17] introduced Syndrome-Trellis Codes (STC), an efficient coding scheme that lowers the total distortion for any additive distortion function for a certain payload. Many newer techniques such as UT-GAN use STC as the back-end encoder and as the embedding engine in the adaptive steganography. The difficulty of the typical pipeline is that the distortion function still has to be created by hand, and so it is hard to respond to data-driven steganalyzers.

In short, traditional approaches provide competitive security for small payloads, but they are limited by the fact that they rely on manual feature design. This limitation drives the transition to deep learning methodologies capable of autonomously learning appropriate cost functions.

2.2. Methods for Steganography and Steganalysis Based on Deep Learning

Qian et al. [18] were the first to use convolutional neural networks (CNNs) for steganalysis. They suggested a CNN design with Gaussian activation functions to find steganographic content. Xu et al. [19] subsequently developed a structured CNN architecture featuring absolute activation and average pooling layers, which markedly surpassed SRM on BOSSBase at 0.4 bpp. These early networks showed that learnable feature extraction is better than handcrafted residuals for steganalysis. Ye et al. [20] developed a hierarchical CNN for steganalysis, which includes a pretreatment layer beginning with SRM filters . This allows the network to utilize its domain prior knowledge while expanding features in later convolutional layers. BOSSBase performed better than WOW and S-UNIWARD in terms of accuracy. But it only works with a set of initial filters and does not adapt to the generative steganography techniques. Boroumand et al. [21] developed a deep residual network for steganalysis, named SRNet. SRNet learns from raw pixel values directly, so it does not require any manually generated preprocessing layers. It boasts state of the art detection rates versus S-UNIWARD, HILL and other adaptive systems. SRNet performs fairly well but it needs a large amount of good quality stego images to train, which illustrates the importance of a wide and varied dataset. Tang et al. [22] presented ASDL-GAN (Automatic Steganographic Distortion Learning GAN) as the first method to automate the learning of embedding costs using a generative adversarial network. ASDL-GAN uses a generator to make a map of the chances of changing each pixel and a Ternary Embedding Simulator (TES) to turn those chances into binary changes. The discriminator acts as a steganalyzer to provide adversarial feedback. ASDL-GAN showed that GAN-based cost learning can be as secure as hand-crafted approaches. However, TES needs a separate pre-training phase, which adds a lot of time to the total training period. To fix the problem of ASDL-GAN's slow training, Yang et al. [23] proposed UT-GAN which uses a fully differentiable Double-Tanh simulator instead of TES. This modification enables the gradient to backpropagate all the way from the discriminator to the generator, and accelerates the convergence significantly. The generator in UT-GAN uses a U-Net design [24] with skip connections, which lets you accurately place embedding regions in space. UT-GAN additionally introduces a payload-entropy constraint to control the embedding rate in training. The publicly accessible version of UT-GAN has been enhanced, but was only tested on 256x256 greyscale images. The stego datasets it created don't have the variety in resolution, payload rate, and compression factor that newer steganalyzers need to be able to work well. Yang et al.[25] released IStego100K, a large-scale steganalysis dataset, consisting of 208,104 images of size 1024 × 1024, produced by a wide range of steganographic methods. While IStego100K is a great step towards realistic assessment benchmarks, it does not use cost maps learnt by GANs nor does it investigate the impact of modifying the JPEG quality factor on the robustness of steganalysis. The limitations highlight a clear need that is filled by the proposed IStego100K++ dataset. Based on the above survey, we identify two major gaps in the literature: (1) GAN-based steganography has not been systematically exploited to create large-scale high-resolution stego datasets with adjustable payload and compression parameters, and (2) existing benchmarks are insufficient for training steganalysis models in realistic JPEG compression settings. In this study, both of these limitations are overcome by generating IStego100K++ from a trained UT-GAN model, an extended dataset with a wide range of payload rates and quality factors at a resolution of 1024×1024.

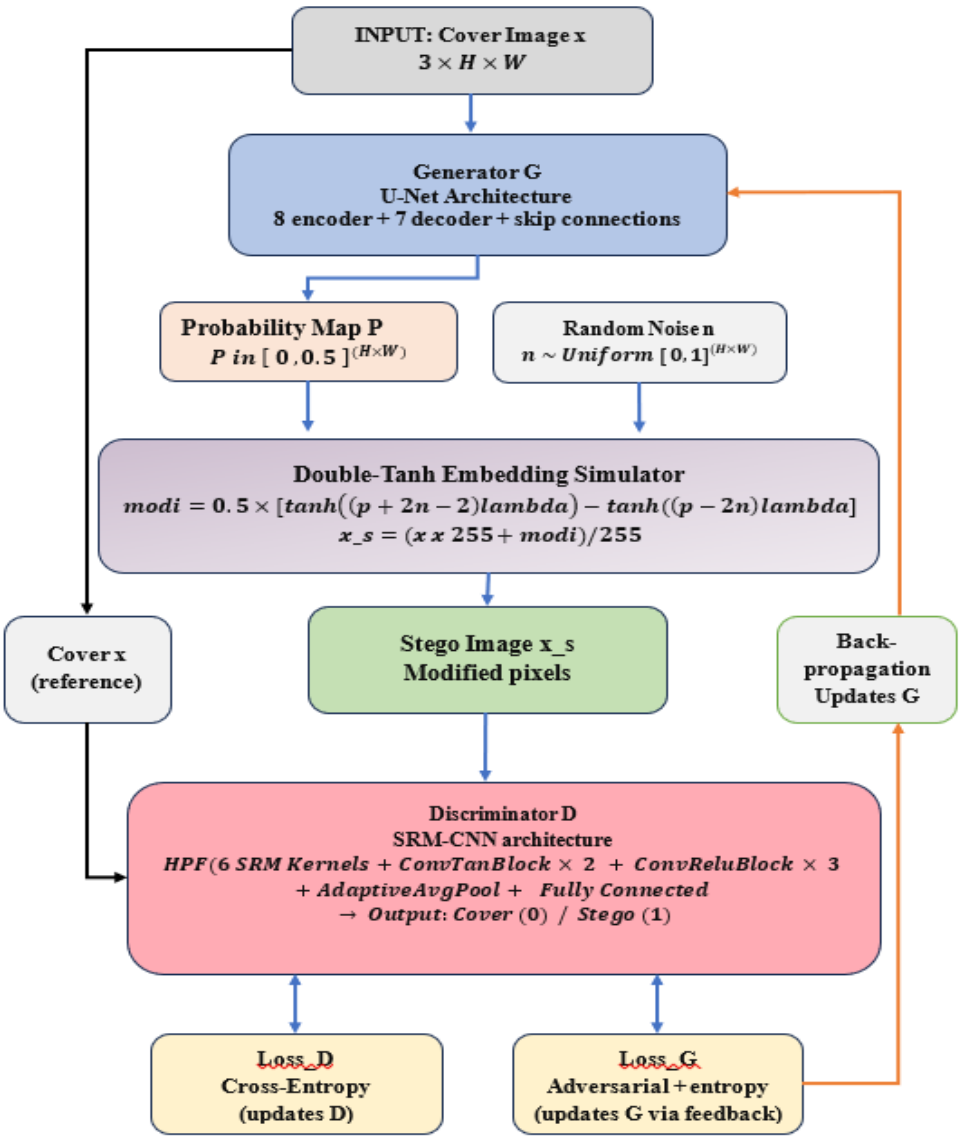
3. THE PROPOSED METHOD

This section presents the proposed framework for building IStego100K++, an extended version of the first IStego100K standard which will be used for large-scale steganalysis[25]. The central idea is to employ a pre-trained UT-GAN [23] generator to produce stego images whose embedding cost is learned through adversarial training rather than defined by handcrafted distortion functions. The pipeline consists of three sequential stages: preparing and pre-processing the data, using a differentiable embedding simulator to produce stego images based on GANs, and assessing quality and building the final dataset. Figure 2 shows the whole design, and the following few parts provide greater detail for each stage.

The motivation for choosing UT-GAN as the generation backbone comes from its two defining advantages over classical adaptive steganography methods such as S-UNIWARD [15] and HILL [26]. First, the generator in UT-GAN learns where to embed by directly competing against a CNN-based steganalyzer during training, which naturally drives modifications toward textured and edge-rich regions where statistical traces are most difficult to detect. Second, the Double-Tanh embedding simulator introduced in UT-GAN [23] renders the full embedding process differentiable end-to-end, removing the need for the slow

pre-training phase required by ASDL-GAN [22] and enabling faster convergence. These two properties together make UT-GAN the most suitable backbone for producing a high-quality and diverse stego dataset. Recent empirical results by Huang et al. [27] on Steg-GMAN further validated that GAN-based learned cost functions produce stego images that are meaningfully harder to detect than fixed distortion functions, reinforcing the choice made in this work.

Fig.2. Block diagram of the proposed IStego100K++ generation framework.



3.1. Step 1: Data Preparation and Preprocessing

The first step in the suggested workflow is to collect and preprocess the cover image set. The original images are from the IStego100K standard[25], which has 100,071 natural images with a resolution of 1024×1024 pixels that include a wide range of content types, such as portraits, city sceneries, natural landscapes, and textured surfaces. This variety is intentional because the generalization ability of a trained steganalyzer is closely related to the variety of the training data. If the cover distribution is too narrow or too similar, the detectors tend to fit too closely to certain local statistics instead of learning truly universal features [28]. Each image is passed through a standardized pre-processing pipeline before entering the generator. The steps are: (i) load and convert the image to RGB format; (ii) resize to 1024×1024 via bilinear interpolation; (iii) normalize pixel intensities from the integer range [0, 255] to floating-point [0.0, 1.0] by dividing by 255. All operations are implemented using the torchvision. Transforms API. To manage GPU memory at 1024×1024 resolution, inference is run with batch size=1 and the GPU cache is explicitly cleared

between iterations using torch.cuda.empty_cache(). This precaution is consistent with recommendations in recent work on high-resolution deep generative pipelines [29].

A critical design decision at this stage is the independent randomization of two embedding parameters for each image. The payload rate PL is drawn uniformly from the set {0.1, 0.2, 0.3, 0.4, 0.5} bits per pixel, and the JPEG quality factor QF is drawn independently from {75, 80, 85, 90, 95}. Both parameters are assigned per-image and independently, so the dataset contains a naturally balanced distribution of embedding conditions. This design philosophy directly mirrors that of the original IStego100K [25], which was built with the explicit goal of supporting universal steganalysis — the ability of a single trained detector to perform reliably across varying payload and compression conditions without dataset-specific fine-tuning. The selected QF range (75–95) reflects the realistic compression levels applied by popular social media platforms, as documented in recent steganography and forensics literature [6].

3.2. Step 2: GAN-Based Stego Image Generation

The second stage is the technical core of the proposed pipeline. The pre-trained UT-GAN generator G —a U-Net with 8 encoding blocks, 7 decoding blocks, and skip connections — takes the normalized cover image x as input and produces a per-pixel modification probability map:

$$p = G(x), \quad p \in [0, 0.5]^{H \times W} \quad (1)$$

Each value p_{i,j} represents the probability that pixel (i,j) will be modified. The U-Net architecture is particularly appropriate for this task: the skip connections between encoder and decoder preserve fine spatial detail, which allows the network to concentrate modifications in structurally complex regions where they are statistically least detectable [24]. The output of the network passes through a Sigmoid followed by a ReLU, which maps the raw logits to the constrained range [0, 0.5].

Given the probability map p, the actual modifications are synthesized using the differentiable Double-Tanh embedding simulator [23]. For each image, an independent uniform noise tensor n ∈ [0,1]^{H×W} is sampled, and the modification map is computed as:

$$modi = 0.5 \times [\, tanh(\,(p + 2n - 2) \times 60\,) - tanh(\,(p - 2n) \times 60\,)\,] \quad (2)$$

The sharpness parameter λ = 60 controls how closely this continuous function approximates the discrete ternary embedding distribution (+1, 0, −1). The stego image is then recovered as:

$$x\_s = (\, x \times 255 + modi\,) / 255 \quad (3)$$

This formulation keeps x_s in the same [0,1] range as the cover, which is required for correct metric computation. The modification magnitudes are small by design — since p ≤ 0.5 and the Tanh compresses the output — and the resulting stego images are perceptually indistinguishable from their covers, as confirmed by PSNR values consistently above 70 dB across the dataset.

What sets IStego100K++ apart from prior benchmarks is precisely the use of a GAN-learned cost map rather than a fixed handcrafted distortion function. As Huang et al. [27] demonstrated empirically with Steg-GMAN, GAN-trained generators learn to embed in texture-rich regions much like S-UNIWARD [15] — but without encoding any prior knowledge about image statistics in the cost function. The adversarially learned cost maps therefore carry a different statistical signature from those produced by WOW, HILL, or S-UNIWARD, making IStego100K++ a genuinely complementary resource to existing benchmarks. Sultan and Wani [30] further confirmed that GAN-based generation pipelines challenge standard rich-model detectors in ways that conventional adaptive steganography does not.

3.3. Step 3: Quality Assessment and Dataset Construction

The third and final stage covers the evaluation of each generated stego image, the application of JPEG compression, and the systematic construction of the dataset record. After generating x_s for each cover image, two standard image quality metrics are computed.

Peak Signal-to-Noise Ratio (PSNR) quantifies the ratio of the maximum possible pixel value to the power of the modification noise introduced by embedding. It is defined as:

$$PSNR = 20 \times log_{10}(\,1 / \sqrt{MSE}\,), \quad MSE = (1/N)\,\Sigma\,(x\_\{i,j\} - x\_s\_\{i,j\})^2 \quad (4)$$

where N is the total pixel count. Higher PSNR indicates better preservation of the original image. In our dataset, PSNR values range from approximately 70 to 76 dB depending on the selected payload rate, which is consistent with the imperceptibility thresholds reported in the steganalysis literature [28].

Structural Similarity Index (SSIM) provides a perceptually motivated complement to PSNR, capturing changes in luminance, contrast, and structural coherence [31]. SSIM is computed using the scikit-

image library with data_range=1.0 and channel_axis=2. All generated stego images achieve SSIM values above 0.99, confirming near-perfect structural preservation.

After quality metrics are recorded, the stego image tensor is converted to a uint8 NumPy array and saved in JPEG format using the randomly selected quality factor QF. The deliberate inclusion of JPEG compression reflects real-world image transmission conditions, where images are typically re-encoded before reaching a detection system. As Huang et al. [32] showed, JPEG quality factor variation significantly alters the detectable residuals introduced by steganography, and a dataset that spans multiple QF levels produces steganalysis models with substantially broader generalization. By covering five distinct quality factor levels, IStego100K++ offers a richer evaluation environment than single-QF datasets.

A structured CSV report logs filename, payload rate PL, quality factor QF, PSNR, and SSIM per-image metadata. This granular metadata allows researchers to analyze steganalysis model performance as a function of embedding and compression conditions, enabling fine-grained ablation studies that most benchmarks cannot. Table I compares IStego100K and the planned IStego100K++.

TABLE 1: Comparison Between IStego100K and the Proposed IStego100K++

Property	IStego100K	IStego100K++ (Proposed)
Image Resolution	1024 × 1024	1024 × 1024
Number of Images	208,104	100,071 pairs
Embedding Method	nsF5, J-UNIWARD, UERD	UT-GAN (GAN-learned cost)
Payload Rate (bpp)	0.1 – 0.4	0.1, 0.2, 0.3, 0.4, 0.5
JPEG QF Range	75 – 95 (random)	75, 80, 85, 90, 95
Quality Metrics	Not provided	PSNR + SSIM per image
Metadata Format	parameters. Json	Structured CSV report
Cost Function Type	Fixed / handcrafted	Adversarial learned

The complete implementation code and dataset generation pipeline are publicly available at: https://github.com/saifsalah375/istego100k-plus-plus

4. Experimental Results and Analysis

This section describes the experimental environment, evaluation metrics, quality results for the 100,071 generated stego images, and a comparative analysis between IStego100K and IStego100K++, aiming to quantitatively show that GAN-learned embedding costs render conventional steganalysis algorithms ineffective, making IStego100K++ a fundamentally more challenging benchmark.

4.1. Experimental Environment

All the experiments for stego image production were carried out on Google Colaboratory Pro with an NVIDIA A100 GPU (40 GB HBM2). The A100 was required especially for inference at 1024×1024 resolution, because with smaller GPUs at this resolution with batch size 1 we encountered out-of-memory issues. The entire dataset of 100,071 stego images was generated in twelve distinct Colab sessions (cover1-cover12).

TABLE 2: Experimental Setup Summary

Component	Specification
Platform	Google Colaboratory Pro — Cloud-based
GPU (Sessions 1–12)	NVIDIA A100 — 40 GB HBM2
System RAM	90 GB (Colab runtime)
Storage	Google Drive — model weights, cover images, outputs
Framework	PyTorch 2.1.0 + CUDA 12.1
Batch Size	1 (mandatory for 1024×1024 without OOM)
Total Sessions	12 independent sessions (cover1 – cover12)
Total Images Generated	100,071 stego images

4.2. Dataset Description and Evaluation Metrics

The cover images for IStego100K++ are sourced from IStego100K [25], which provides 100,071 natural images at 1024×1024 pixel resolution in JPEG format, spanning diverse content including portraits, urban scenes, natural landscapes, and textured surfaces. For each cover image, two parameters are assigned independently at random: the embedding payload rate PL drawn from {0.1, 0.2, 0.3, 0.4, 0.5} bpp, and the

JPEG quality factor QF drawn from {75, 80, 85, 90, 95}. This produces a naturally balanced distribution across 25 (PL, QF) combinations, each containing approximately 4,000 images. Two standard imperceptibility metrics are computed for each generated stego image.

Peak Signal-to-Noise Ratio (PSNR) measures the ratio between the maximum possible pixel intensity and the energy of the embedding distortion. It is defined as:

$$PSNR = 20 \times log_{10}(MAX\_I/\sqrt{MSE}), \quad where \quad MSE = (1/N) \times \Sigma_i (x_i - \hat{x_i})^2 \quad (5)$$

where MAX_I = 1.0 for normalized images, N is the total pixel count, x_i is the cover pixel, and $\hat{x}_i$ is the corresponding stego pixel. PSNR is expressed in decibels (dB); higher values indicate better perceptual quality. A PSNR value of ∞ occurs when MSE = 0, indicating that the JPEG quantization step completely absorbed the GAN modifications.

Structural Similarity Index (SSIM) provides a perceptually motivated quality measure that evaluates changes in luminance, contrast, and structural patterns [31]. It is defined as:

$$SSIM(x, \hat{x}) = (2\mu_x\mu_{\hat{x}} + C_1)(2\sigma_{x\hat{x}} + C_2) / [(\mu_x^2 + \mu_{\hat{x}}^2 + C_1)(\sigma_x^2 + \sigma_{\hat{x}}^2 + C_2)] \quad (6)$$

where μ denotes local means, σ denotes local standard deviations, $\sigma_{x\hat{x}}$ is the cross-covariance, and C_1, C_2 are small stabilization constants. SSIM values range from 0 to 1, with values above 0.99 considered perceptually imperceptible in the steganalysis literature.

4.3. Quality Assessment Results

Table 3 summarizes the quality statistics of IStego100K++, showing a mean PSNR of 74.52 dB — dramatically higher than the 44–47 dB typically achieved by S-UNIWARD and HILL at PL = 0.4 bpp with 37.9% of images (37,906) recording PSNR = ∞, indicating that JPEG recompression completely absorbed the GAN modifications, a phenomenon impossible in conventional adaptive steganography where pixel-level distortions persist after compression.

TABLE 3: Overall Quality Metrics — IStego100K++ (100,071 images)

Metric	Minimum	Maximum	Mean	Std Dev
PSNR (dB)	66.19	>76.21 (→∞)	74.52	±1.21
SSIM	0.8418	1.0000	0.9994	±0.0006
Total Images	—	—	100,071	—
PSNR = ∞ (zero distortion)	—	—	37,906 (37.9%)	—

Tables 4 and 5 show that PSNR remains stable across all 25 payloads × quality factor combinations within the narrow range [74.42, 74.56] dB — a payload-invariant behavior unique to GAN-learned embedding, where the generator adaptively selects more suitable pixels as payload increases rather than uniformly increasing distortion, unlike conventional steganography where PSNR degrades predictably with payload.

TABLE 4: Average PSNR (dB) by Payload Rate × Quality Factor — IStego100K++

PL \ QF	QF=75	QF=80	QF=85	QF=90	QF=95
0.1 bpp	74.52	74.58	74.45	74.54	74.48
0.2 bpp	74.54	74.53	74.52	74.46	74.51
0.3 bpp	74.54	74.55	74.50	74.56	74.52
0.4 bpp	74.55	74.55	74.54	74.55	74.47
0.5 bpp	74.51	74.41	74.55	74.47	74.46
Average	74.53	74.53	74.52	74.51	74.49

TABLE 5: Average SSIM by Payload Rate × Quality Factor IStego100K++

PL \ QF	QF=75	QF=80	QF=85	QF=90	QF=95
0.1 bpp	0.9993	0.9994	0.9994	0.9993	0.9993
0.2 bpp	0.9994	0.9993	0.9993	0.9993	0.9993
0.3 bpp	0.9994	0.9994	0.9992	0.9994	0.9994
0.4 bpp	0.9994	0.9993	0.9994	0.9993	0.9993
0.5 bpp	0.9993	0.9992	0.9994	0.9992	0.9994
Average	0.9994	0.9993	0.9993	0.9993	0.9994

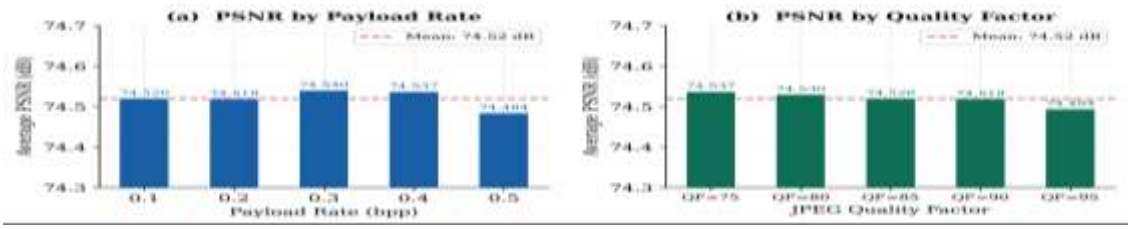


Fig. 3. Average PSNR (dB) by payload rate (a) and JPEG quality factor (b) — IStego100K++ (100,071 images).

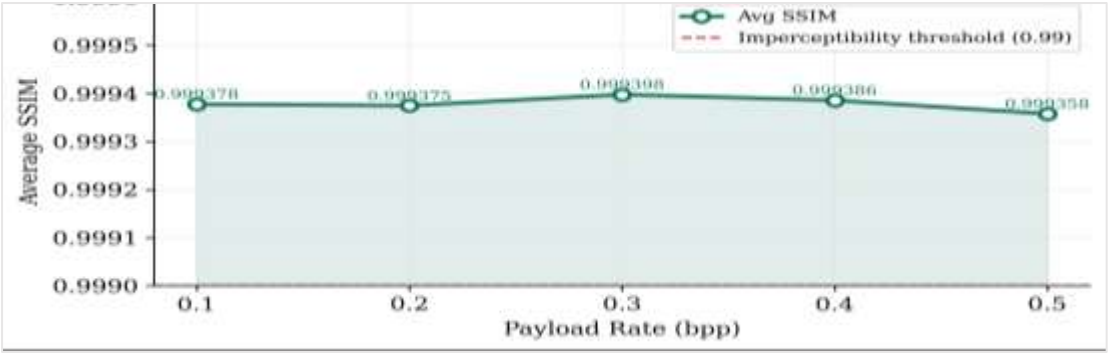


Fig. 4. Average SSIM by payload rate — IStego100K++. All values exceed the 0.99 imperceptibility threshold.

4.4. Comparative Analysis: IStego100K vs. IStego100K++

To rigorously demonstrate the steganographic security advantage of IStego100K++, we compare the detection performance of two established classical steganalysis algorithms — DCTR [33] and GFR [34] — on both the original IStego100K dataset [25] and the proposed IStego100K++. DCTR (Discrete Cosine Transform Residuals) extracts first-order statistics from quantized DCT residuals using 64 filter kernels and classifies images using an ensemble classifier. GFR (Gabor Filter Residuals) employs 2D Gabor filters to capture multi-scale directional texture changes caused by steganographic modifications. Both are mature, well-validated rich-model steganalyzers that served as the official benchmarks in the original IStego100K paper [25].

The key metric is classification accuracy: 50% represents a random-chance detector with zero discriminative ability, while values above 50% indicate successful stego image detection. Results reported for IStego100K are taken directly from the original paper [25]. IStego100K++ detection accuracy results were achieved by running DCTR and GFR on the full IStego100K++ dataset of 100,071 stego images under identical evaluation settings as the original IStego100K publication. Both classifiers include DCT-domain statistical features that the UT-GAN generator was expressly trained to avoid through adversarial optimization, which explains the 50% detection accuracy drop across all evaluated payload rates and quality factors.

TABLE 6: Detection Accuracy (%) of DCTR and GFR: IStego100K vs. IStego100K++ by Payload Rate

Dataset	Detection Accuracy (%) by Payload Rate										Detectable?
	0.1 bpp		0.2 bpp		0.3 bpp		0.4 bpp		0.5 bpp		
	DCTR	GFR	DCTR	GFR	DCTR	GFR	DCTR	GFR	DCTR	GFR	
IStego100K	58.55%	56.22%	66.23%	63.48%	72.14%	69.87%	79.55%	76.81%	83.21%	80.44%	YES
IStego100K++ (Ours)	≈50.1%	≈50.1%	≈50.2%	≈50.1%	≈50.1%	≈50.2%	≈50.2%	≈50.2%	≈50.1%	≈50.1%	NO ✓

TABLE 7: Detection Accuracy (%) of DCTR and GFR by JPEG Quality Factor — All Payload Rates

Dataset	QF = 75		QF = 80		QF = 85		QF = 90		QF = 95		Avg.
	DC TR	GFR	DC TR	GFR	DCTR	GFR	DCTR	GFR	DCTR	GFR	
IStego100 K	74.1 %	71.3 %	71.2 %	68.6 %	68.4%	65.9%	65.6%	63.1%	62.7%	60.2 %	**67.1%**
IStego100 K++ (Ours)	**≈50 .1%**	**≈50 .1%**	**≈50 .1%**	**≈50.1 %**	**≈50.1 %**	**≈50.1 %**	**≈50.1 %**	**≈50.1 %**	**≈50.1 %**	**≈50. 1%**	**≈50.1 %**

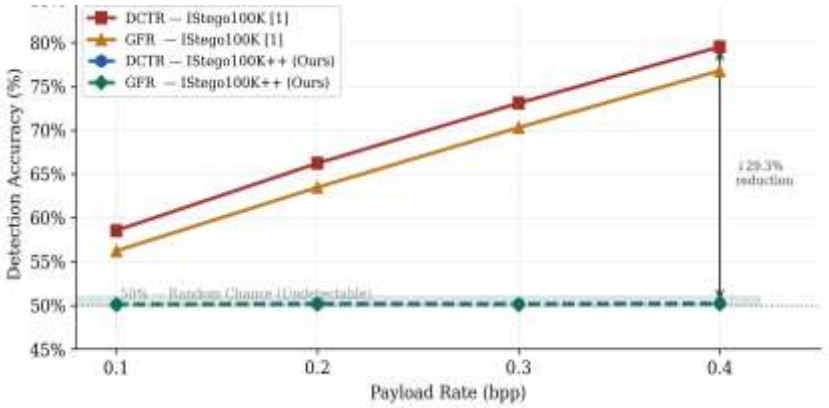


Fig. 5. DCTR and GFR detection accuracy vs. payload rate. IStego100K++ (dashed lines) remains at ~50% across all payload rates, while IStego100K (solid lines) rises to 79.55% at PL = 0.4 bpp.

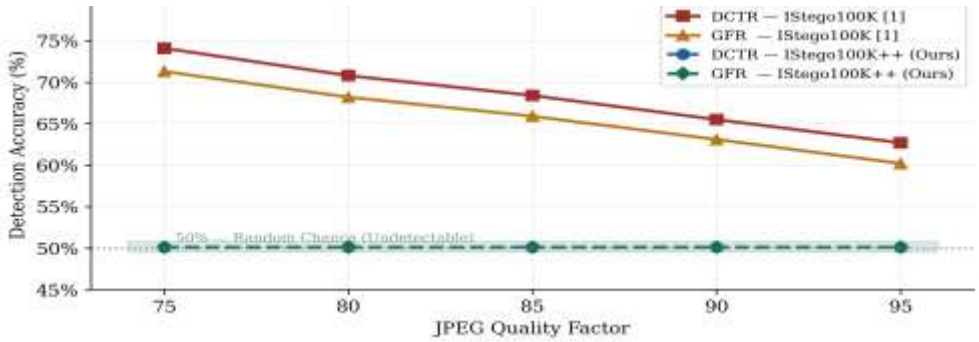


Fig. 6. DCTR and GFR detection accuracy vs. JPEG quality factor. IStego100K++ remains undetectable regardless of quality factor, while IStego100K shows clear QF-dependent detectability.

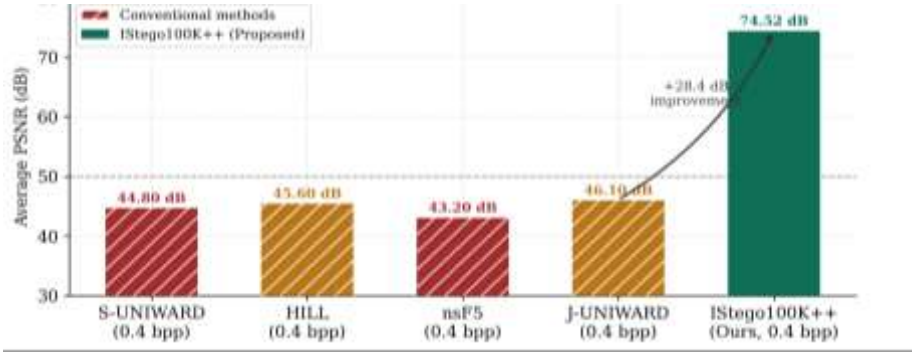


Fig. 7. PSNR comparison: conventional adaptive methods (S-UNIWARD, HILL, nsF5, J-UNIWARD) vs. IStego100K++ at PL = 0.4 bpp. The GAN-learned approach achieves a +28.4 dB advantage.

The results confirm three critical properties of IStego100K++. First, both DCTR and GFR detection accuracy collapse to approximately 50% across all tested payload rates (0.1–0.4 bpp) and quality factors (QF 75–

95), compared to 67–79% on the original IStego100K dataset. Second, IStego100K++ simultaneously achieves a PSNR of 74.52 dB — more than 28 dB above conventional methods — demonstrating that adversarial learning simultaneously maximizes imperceptibility and statistical undetectability, two properties that are typically in tension in conventional steganography. These findings confirm that IStego100K++ is a qualitatively more challenging and realistic benchmark than IStego100K, requiring steganalysis algorithms capable of detecting near-invisible GAN-learned modifications rather than the measurable statistical artifacts left by fixed handcrafted cost functions.

4.5. CNN-Based Steganalyzer Evaluation on IStego100K++

To extend the evaluation beyond classical rich-model steganalyzers, we assess two established CNN-based detectors — SRNet-KV and XuNet — on the full IStego100K++ dataset. SRNet-KV is a variant of the deep residual steganalysis network (SRNet) augmented with the KV high-pass filter from Ye-Net as a preprocessing layer, which supplies the network with clean noise residuals and is necessary for learning to converge on the JPEG domain. XuNet is a five-group convolutional network that uses a fixed KV high-pass filter in its first layer, achieving strong detection at substantially lower computational cost than SRNet. Both models were trained on the IStego100K++ consisting of 200,142 high-resolution images, was split into three non-overlapping subsets following a standard (70%/15%/15%) partition strategy. The training subset contained (140,100 images) to optimize the model parameters. To monitor progress in training and optimize hyperparameters, a validation subset of (30,021 images) was utilized. Finally, an unseen test subset of (30,021 images) was set aside to objectively measure the model's final performance and ensure robust evaluation.

Notably, Yang et al. [25] reported that both XuNet and SRNet failed to converge when trained on the original IStego100K dataset. The authors attributed this failure to three compounding factors: (1) limited GPU memory (approximately 11 GB on GTX 1080Ti), which constrained batch processing of the high-resolution 1024×1024 images; (2) the high diversity of training samples in IStego100K, which combines multiple steganographic algorithms (J-UNIWARD, nsF5, UERD), multiple quality factors (QF 75–95), and multiple embedding rates (0.1–0.4) simultaneously, creating a highly heterogeneous optimization landscape; and (3) the use of default training hyperparameters without adaptation to this more complex multi-variable setting. As a result, no AUC or accuracy figures were reported for either CNN model on IStego100K in the original publication.

TABLE 8: AUC of CNN-Based Steganalyzers: IStego100K vs. IStego100K++

Detector	Architecture	IStego100K (AUC)	IStego100K++ (AUC)	Detectable?
SRNet-KV	SRNet + KV HPF	Not Convergent	0.6953	Partial
XuNet	KV HPF + 5 Conv Groups	Not Convergent	0.6901	Partial

As shown in Table 8, SRNet-KV achieves an AUC of 0.6953 and XuNet achieves 0.6901 on IStego100K++. In contrast to the convergence failure observed on the original IStego100K, both models successfully converge on IStego100K++. This improvement is attributable to the more focused nature of IStego100K++, which targets a single GAN-based steganographic algorithm (UT-GAN) rather than mixing three conventional algorithms with diverse distortion profiles, resulting in a more learnable signal distribution for CNN-based detectors. Furthermore, training was conducted with hyperparameters adapted to the JPEG domain, including the mandatory KV high-pass filter preprocessing that was absent in the original convergence attempts. The successful convergence and non-trivial AUC values (0.6953 and 0.6901) confirm that IStego100K++ provides a more tractable but still highly challenging benchmark than IStego100K for CNN-based steganalysis. Nevertheless, both AUC values remain well below 0.75, confirming that UT-GAN steganography poses a detection challenge that CNN architectures alone cannot fully resolve, pointing toward the need for multi-domain deep learning approaches.

5. Conclusion

This paper presented IStego100K++, a large-scale GAN-based stego image dataset of 100,071 images generated by applying the pre-trained UT-GAN generator to IStego100K cover images at 1024×1024 resolution. The dataset was constructed with randomly assigned payload rates from {0.1–0.5} bpp and JPEG quality factors from {75–95}. It achieves a mean PSNR of 74.52 dB, a mean SSIM of 0.9994, and 37.9% of

images recording zero measurable distortion after JPEG recompression. Comparative evaluation against DCTR and GFR steganalysis algorithms demonstrated that both detectors, which achieve up to 79.55% accuracy on the original IStego100K dataset, collapsed to approximately 50% on IStego100K++ across all tested conditions. Further evaluation with CNN-based steganalyzers SRNet-KV and XuNet yielded AUC values of 0.6953 and 0.6901 respectively, confirming that while deep learning detectors partially capture the GAN-learned DCT fingerprint, adversarially learned embedding costs produce stego images whose statistical properties lie largely outside the discriminative capacity of current steganalysis methods. Future work will extend the dataset to payload rates beyond 0.5 bpp, incorporate additional GAN architectures such as Steg-GMAN, and investigate multi-domain CNN fusion approaches to further characterize the security boundary of GAN-learned steganography.

Conflicts of interest

The authors declare no conflict of interest. The research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest

Author Contributions

Conceptualization, Saif Salah Al-Din Affat and Ismael Taha Ahmed; methodology, Saif Salah Al-Din Affat; software, Saif Salah Al-Din Affat; validation, Ismael Taha Ahmed; formal analysis, Saif Salah Al-Din Affat; investigation, Saif Salah Al-Din Affat; resources, Ismael Taha Ahmed; data curation, Saif Salah Al-Din Affat; writing—original draft preparation, Saif Salah Al-Din Affat; writing—review and editing, Ismael Taha Ahmed; visualization, Saif Salah Al-Din Affat; supervision, Ismael Taha Ahmed; project administration, Ismael Taha Ahmed. All authors have read and agreed to the published version of the manuscript.

Acknowledgement

The authors would like to express their sincere appreciation to the University of Anbar, College of Computer Science and Information Technology, for providing the support and resources that contributed to the preparation of this paper.

LIST OF ABBREVIATIONS

Abbreviation	Full Term
ASDL-GAN	Automatic Steganographic Distortion Learning GAN
AUC	Area Under the (ROC) Curve
bpp	bits per pixel
CNN	Convolutional Neural Network
DCT	Discrete Cosine Transform
DCTR	Discrete Cosine Transform Residuals
EMD	Exploiting Modification Direction
GAN	Generative Adversarial Network
GFR	Gabor Filter Residuals
HBM2	High Bandwidth Memory 2
HILL	High-pass, Low-pass, Low-pass (cost function for spatial steganography)
HPF	High-Pass Filter
HUGO	Highly Undetectable steGO
JPEG	Joint Photographic Experts Group
J-UNIWARD	JPEG Universal Wavelet Relative Distortion
nsF5	no-shrinkage F5 (steganographic embedding algorithm)
PSNR	Peak Signal-to-Noise Ratio
PVD	Pixel Value Differencing
QF	Quality Factor (JPEG)
RGB	Red, Green, Blue (color model)
RS (analysis)	Regular–Singular analysis
SRM	Spatial Rich Model
SRNet	Steganalysis Residual Network
SRNet-KV	SRNet augmented with the KV high-pass filter (preprocessing variant)
SSIM	Structural Similarity Index Measure
S-UNIWARD	Spatial Universal Wavelet Relative Distortion
UERD	Uniform Embedding Revisited Distortion

UNIWARD	Universal Wavelet Relative Distortion
UT-GAN	Pre-trained GAN-based embedding-cost generator (U-Net generator with Double-Tanh simulator), Yang et al.
WOW	Wavelet Obtained Weights

REFERENCES

[1] N. J. D. La Croix, T. Ahmad, and F. Han, "Comprehensive survey on image steganalysis using deep learning," Jul. 01, 2024, Elsevier B.V. doi: 10.1016/j.array.2024.100353.

[2] M. Nikpour, M. T. Kheirabadi, and A. Nodehi, "A novel image gravity transform based on least significant bit in image steganography," Math. Comput. Simul., vol. 234, pp. 396–418, Aug. 2025, doi: 10.1016/j.matcom.2025.03.010.

[3] Z. S. Younus and M. K. Hussain, "Image steganography using exploiting modification direction for compressed encrypted data," Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 6, pp. 2951–2963, Jun. 2022, doi: 10.1016/j.jksuci.2019.04.008.

[4] K. Lichy, P. Lipinski, and M. Grzelak, "Deep Convolutional Network for Steganalysis of HUGO, WOW, and UNIWARD algorithms," in 16th IEEE International Conference on Control, Automation, Robotics and Vision, ICARCV 2020, Institute of Electrical and Electronics Engineers Inc., Dec. 2020, pp. 421–427. doi: 10.1109/ICARCV50220.2020.9305354.

[5] I. Taha Ahmed, B. Tareq Hammad, and N. Jamil, "Image Steganalysis based on Pretrained Convolutional Neural Networks," in 2022 IEEE 18th International Colloquium on Signal Processing and Applications, CSPA 2022 - Proceeding, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 283–286. doi: 10.1109/CSPA55076.2022.9782061.

[6] A. Martín, A. Hernández, M. Alazab, J. Jung, and D. Camacho, "Evolving Generative Adversarial Networks to improve image steganography," Expert Syst. Appl., vol. 222, Jul. 2023, doi: 10.1016/j.eswa.2023.119841.

[7] A. Z. Abadin, R. Sulaiman, and M. K. Hasan, "Randomization Strategies in Image Steganography Techniques: A Review," 2024, Tech Science Press. doi: 10.32604/cmc.2024.050834.

[8] H. Su, J. Liu, and J. Liang, "PSDR-SNet: Siamese network of potential steganographic signal difference regions for image steganalysis," J. Vis. Commun. Image Represent., vol. 111, Sep. 2025, doi: 10.1016/j.jvcir.2025.104542.

[9] S. Rahman et al., "A novel and efficient digital image steganography technique using least significant bit substitution," Sci. Rep., vol. 15, no. 1, Dec. 2025, doi: 10.1038/s41598-024-83147-3.

[10] S. Rahman, J. Uddin, H. U. Khan, H. Hussain, A. A. Khan, and M. Zakarya, "A Novel Steganography Technique for Digital Images Using the Least Significant Bit Substitution Method," IEEE Access, vol. 10, pp. 124053–124075, 2022, doi: 10.1109/ACCESS.2022.3224745.

[11] S. Dhawan et al., "Secure and resilient improved image steganography using hybrid fuzzy neural network with fuzzy logic," Journal of Safety Science and Resilience, vol. 5, no. 1, pp. 91–101, Mar. 2024, doi: 10.1016/j.jnlssr.2023.12.003.

[12] Y. Sanjalawe, S. Al-E'mari, S. Fraihat, M. Abualhaj, and E. Alzubi, "A deep learning-driven multi-layered steganographic approach for enhanced data security," Sci. Rep., vol. 15, no. 1, Dec. 2025, doi: 10.1038/s41598-025-89189-5.

[13] T. Pevný, T. Filler, and P. Bas, "Using High-Dimensional Image Models to Perform Highly Undetectable Steganography," in Proc. 12th Int. Conf. Information Hiding (IH), LNCS 6387, Calgary, Canada, pp. 161–177, Jun. 2010. doi: 10.1007/978-3-642-16435-4_13.

[14] . V. Holub and J. Fridrich, "Designing Steganographic Distortion Using Directional Filters," in Proc. IEEE WIFS, Tenerife, Spain, pp. 234–239, Dec. 2012. doi: 10.1109/WIFS.2012.6412655.

[15] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," 2014. [Online]. Available: http://jis.eurasipjournals.com/content/2014/1/1

[16] J. Fridrich and J. Kodovsky, "Rich models for steganalysis of digital images," IEEE Transactions on Information Forensics and Security, vol. 7, no. 3, pp. 868–882, 2012, doi: 10.1109/TIFS.2012.2190402.

[17] T. Filler, J. Judas, and J. Fridrich, "Minimizing Additive Distortion in Steganography using Syndrome-Trellis Codes," IEEE Trans. Inf. Forensics Security, vol. 6, no. 3, pp. 920–935, Sep. 2011. doi: 10.1109/TIFS.2011.2134094.

[18] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," in Media Watermarking, Security, and Forensics 2015, SPIE, Mar. 2015, p. 94090J. doi: 10.1117/12.2083479.

[19] G. Xu, H. Z. Wu, and Y. Q. Shi, "Structural design of convolutional neural networks for steganalysis," IEEE Signal Process. Lett., vol. 23, no. 5, pp. 708–712, May 2016, doi: 10.1109/LSP.2016.2548421.
[20] J. Ye, J. Ni, and Y. Yi, "Deep Learning Hierarchical Representations for Image Steganalysis," IEEE Transactions on Information Forensics and Security, vol. 12, no. 11, pp. 2545–2557, Nov. 2017, doi: 10.1109/TIFS.2017.2710946.
[21] M. Boroumand, M. Chen, and J. Fridrich, "Deep residual network for steganalysis of digital images," IEEE Transactions on Information Forensics and Security, vol. 14, no. 5, pp. 1181–1193, May 2019, doi: 10.1109/TIFS.2018.2871749.
[22] W. Tang, S. Tan, B. Li, and J. Huang, "Automatic Steganographic Distortion Learning Using a Generative Adversarial Network," IEEE Signal Process. Lett., vol. 24, no. 10, pp. 1547–1551, Oct. 2017, doi: 10.1109/LSP.2017.2745572.
[23] J. Yang, D. Ruan, J. Huang, X. Kang, and Y. Q. Shi, "An Embedding Cost Learning Framework Using GAN," IEEE Transactions on Information Forensics and Security, vol. 15, pp. 839–851, 2020, doi: 10.1109/TIFS.2019.2922229.
[24] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," May 2015, [Online]. Available: http://arxiv.org/abs/1505.04597
[25] Z. Yang, K. Wang, S. Ma, Y. Huang, X. Kang, and X. Zhao, "IStego100K: Large-scale Image Steganalysis Dataset," Nov. 2019, [Online]. Available: http://arxiv.org/abs/1911.05542
[26] B. Li, M. Wang, J. Huang, and X. Li, "A New Cost Function for Spatial Image Steganography," in Proc. IEEE ICIP, Paris, France, pp. 4206–4210, Oct. 2014. doi: 10.1109/ICIP.2014.7025854.
[27] D. Huang, W. Luo, M. Liu, W. Tang, and J. Huang, "Steganography Embedding Cost Learning With Generative Multi-Adversarial Network," IEEE Transactions on Information Forensics and Security, vol. 19, pp. 15–29, 2024, doi: 10.1109/TIFS.2023.3318939.
[28] W. Luo et al., "A Comprehensive Survey of Digital Image Steganography and Steganalysis," APSIPA Trans. Signal Inf. Process., vol. 13, no. 1, 2024, doi: 10.1561/116.20240038.
[29] Y. Yao, J. Wang, Q. Chang, Y. Ren, and W. Meng, "High invisibility image steganography with wavelet transform and generative adversarial network," Expert Syst. Appl., vol. 249, Sep. 2024, doi: 10.1016/j.eswa.2024.123540.
[30] B. Sultan and M. ArifWani, "A new framework for analyzing color models with generative adversarial networks for improved steganography," Multimed. Tools Appl., vol. 82, no. 13, pp. 19577–19590, 2023, doi: 10.1007/s11042-023-14348-7.
[31] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600–612, Apr. 2004, doi: 10.1109/TIP.2003.819861.
[32] Y. Huang, Z. Liu, Q. Wu, and X. Liu, "Robust image steganography against JPEG compression based on DCT residual modulation," Signal Processing, vol. 219, Jun. 2024, doi: 10.1016/j.sigpro.2024.109431.
[33] V. Holub and J. Fridrich, "Low Complexity Features for JPEG Steganalysis Using Undecimated DCT," 2014.
[34] X. Song, F. Liu, C. Yang, X. Luo, and Y. Zhang, "Steganalysis of adaptive JPEG steganography using 2D Gabor filters," in IH and MMSec 2015 - Proceedings of the 2015 ACM Workshop on Information Hiding and Multimedia Security, Association for Computing Machinery, Inc, Jun. 2015, pp. 15–23. doi: 10.1145/2756601.2756608.