
Neural Cover Selection for Image Steganography

Karl Chahine & Hyeji Kim
Department of Electrical and Computer Engineering
University of Texas at Austin
Austin, TX 78712
{karlchahine, hyeji.kim}@utexas.edu

Abstract

In steganography, selecting an optimal cover image—referred to as cover selection—is pivotal for effective message concealment. Traditional methods have typically employed exhaustive searches to identify images that conform to specific perceptual or complexity metrics. However, the relationship between these metrics and the actual message hiding efficacy of an image is unclear, often yielding less-than-ideal steganographic outcomes. Inspired by recent advancements in generative models, we introduce a novel cover selection framework, which involves optimizing within the latent space of pretrained generative models to identify the most suitable cover images, distinguishing itself from traditional exhaustive search methods. Our method shows significant advantages in message recovery and image quality. We also conduct an information-theoretic analysis of the generated cover images, revealing that message hiding predominantly occurs in low-variance pixels, reflecting the waterfilling algorithm’s principles in parallel Gaussian channels. Our code can be found at <https://github.com/karlchahine/Neural-Cover-Selection-for-Image-Steganography>.

1 Introduction

Image steganography embeds secret bit strings within typical cover images, making them imperceptible to the naked eye yet retrievable through specific decoding techniques. This method is widely applied in various domains, including digital watermarking (Cox et al. [2007]), copyright certification (Bilal et al. [2014]), e-commerce (Cheddad et al. [2010]), cloud computing (Zhou et al. [2015]), and secure information storage (Srinivasan et al. [2004]).

Traditionally, hiding techniques such as modifying the least significant bits have been effective for embedding small data volumes up to 0.5 bits per pixel (bpp) (Fridrich et al. [2001]). Leveraging advancements in deep learning, recent approaches employ deep encoder-decoder networks to embed and extract up to 6 bpp, demonstrating significant enhancements in capacity (Chen et al. [2022], Baluja [2017], Zhang et al. [2019]). The encoder takes as input a cover image \mathbf{x} and a secret message \mathbf{m} , outputting a steganographic image \mathbf{s} that appears visually similar to the original \mathbf{x} . The decoder then estimates the message $\hat{\mathbf{m}}$ from \mathbf{s} . The setup is illustrated in Fig. 1 (left).

The effectiveness of steganography is significantly influenced by the choice of the cover image \mathbf{x} , a process known as cover selection. Different images have varying capacities to conceal data without detectable alterations, making cover selection a critical factor in maintaining the reliability of the steganographic process (Baluja [2017], Yaghmaee and Jamzad [2010]).

From a theoretical standpoint, numerous studies have employed information-theoretic analyses to investigate cover selection and determine the capacity limits of information-hiding systems, thereby identifying the maximum number of bits that can be embedded (Moulin et al. [2000], Cox et al. [1999], Moulin and O’Sullivan [2003]). For instance, in Moulin and O’Sullivan [2003], the steganographic

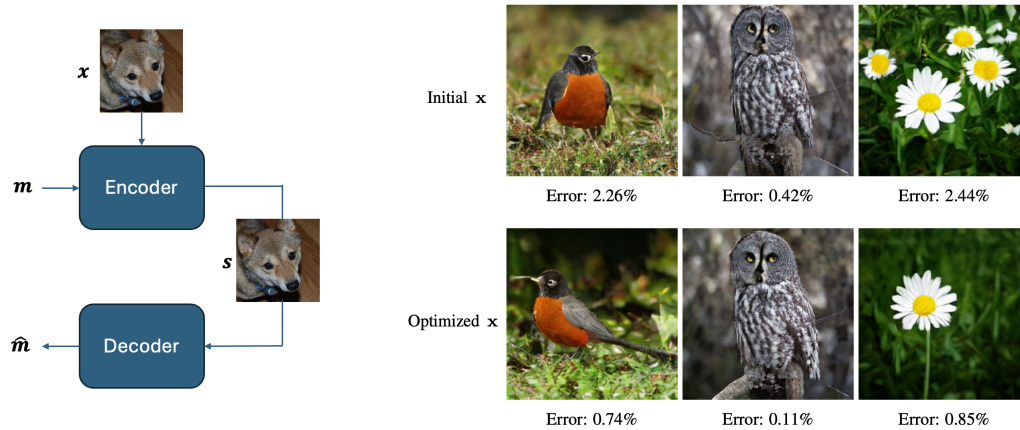


Figure 1: **Left:** Image steganography framework: the encoder takes as input the cover image \mathbf{x} and a secret binary message \mathbf{m} and outputs the steganographic image \mathbf{s} . The decoder then estimates $\hat{\mathbf{m}}$ from \mathbf{s} . **Right:** Randomly sampled cover images from the ImageNet dataset before and after optimization using our framework (described in Section 3). These optimized images demonstrate a significantly reduced error $\|\mathbf{m} - \hat{\mathbf{m}}\|$ while maintaining high image quality.

setup is conceptualized as a communication channel where the cover image \mathbf{x} acts as side information. However, such models are based on impractical assumptions: firstly, the steganographic process is additive—where the message \mathbf{m} is simply added to the cover \mathbf{x} ; and secondly, it presupposes that the cover elements adhere to a Gaussian distribution.

From a practical standpoint, existing techniques for cover selection predominantly rely on exhaustive searches to identify the most suitable cover image. These methods evaluate a variety of image metrics to determine the best candidate from a database. Some strategies include counting modifiable discrete cosine transform (DCT) coefficients to select images with a higher coefficient count for covers (Kharrazi et al. [2006]), assessing visual quality to determine embedding suitability (Evsutin et al. [2018]), and estimating the embedding capacity based on image complexity metrics (Yaghmae and Jamzad [2010], Wang and Zhang [2019]).

Traditional methods for selecting cover images have three key limitations: (i) They rely on heuristic image metrics that lack a clear connection to steganographic effectiveness, often leading to suboptimal message hiding. (ii) These methods ignore the influence of the encoder-decoder pair on the cover image choice, focusing solely on image quality metrics. (iii) They are restricted to selecting from a fixed set of images, rather than generating one tailored to the steganographic task, limiting their ability to find the most suitable cover.

Recent progress in generative models, such as Generative Adversarial Networks (GANs) (Goodfellow et al. [2020]) and diffusion models (Song et al. [2020], Ho et al. [2020]), have ignited significant interest in the area of guided image generation (Shen et al. [2020], Avrahami et al. [2022], Brooks et al. [2023], Gafni et al. [2022], Kim et al. [2022]). Inspired by these innovations, we propose a novel approach that addresses the aforementioned limitations by treating cover selection as an optimization problem.

In our proposed framework, a cover image \mathbf{x} is first inverted into a latent vector, which is then passed through a pretrained generative model to reconstruct the cover image. This image is processed by a neural steganographic encoder to embed a secret message, followed by a decoder to recover the message. We optimize the latent vector to generate an enhanced cover image \mathbf{x}^* , minimizing message recovery errors while preserving the visual and semantic integrity of the image. Fig. 1 (right) presents message recovery errors for randomly selected images before and after optimization. Our approach of optimizing the cover image uncovers a novel way to analyze the transformation from \mathbf{x} to \mathbf{x}^* , revealing that the encoder embeds messages in low-variance pixels, analogous to the water-filling algorithm in parallel Gaussian channels. To the best of our knowledge, this is the first work that examines neural steganographic encoders by framing cover selection as a guided image reconstruction problem.

Our contributions are outlined as follows:

1. **Framework.** We describe the limitations of current cover selection methods and introduce a novel, optimization-driven framework that combines pretrained generative models with steganographic encoder-decoder pairs. Our method guides the image generation process by incorporating a message recovery loss, thereby producing cover images that are optimally tailored for specific secret messages (Section 3).
2. **Experiments.** We validate our methodology through comprehensive experimentation on public datasets such as CelebA-HQ, ImageNet, and AFHQ. Our results demonstrate that the error rates of the optimized images are **an order of magnitude lower** than those of the original images under specific conditions. Impressively, this optimization not only reduces error rates but also enhances the overall image quality, as evidenced by established visual quality metrics. We explore this intriguing phenomenon by examining the correlation between image quality metrics and error rates (Section 3.3).
3. **Interpretation.** We investigate the workings of the neural encoder and find it hides messages within low variance pixels, akin to the water-filling algorithm in parallel Gaussian channels. Interestingly, we observe that our cover selection framework increases these low variance spots, thus improving message concealment (Section 4).
4. **Practical considerations.** We extend our guided image generation process to practical applications, demonstrating its robustness against steganalysis and resilience to JPEG compression, as detailed in Section 5.

Related work. Recent research has explored the use of generative models in steganography. Zhang et al. [2019] introduced a training framework where steganographic encoders and decoders are trained adversarially, similar to GANs. Yu et al. [2024] harness the image translation capabilities of diffusion models to transform a secret image directly into a steganographic image, bypassing the embedding process, a framework known as coverless steganography (Qin et al. [2019]). Shi et al. [2018] is notably relevant, as they created a GAN framework designed to produce images robust against steganalysis. However, there are three key distinctions: (i) they overlooked message error rates, focusing solely on evading detection, compromising the effectiveness of cover images for message recovery; (ii) they trained their GAN from scratch, failing to leverage the advantages of existing pretrained models; and (iii) the images generated were randomly sampled and not user-selectable, limiting application flexibility.

2 Preliminaries

Image steganography aims to hide a secret bit string $\mathbf{m} \in \{0, 1\}^{H \times W \times B}$ into a cover image $\mathbf{x} \in [0, 1]^{H \times W \times 3}$ where the payload B denotes the number of encoded bits per pixel (bpp) and H, W denote the image dimensions. As depicted in Fig. 1 (left), the hiding process is done using a steganographic encoder Enc , which takes as input \mathbf{x} and \mathbf{m} and outputs the steganographic image \mathbf{s} which looks visually identical to \mathbf{x} . A decoder Dec recovers the message, $\hat{\mathbf{m}} = Dec(\mathbf{s})$ with minimal error rate $\frac{\|\mathbf{m} - \hat{\mathbf{m}}\|_0}{H \times W \times B}$.

Cover selection involves generating the ideal cover image \mathbf{x} , to achieve three primary objectives: (i) minimize the error rate as defined above, (ii) ensure that the steganographic image \mathbf{s} visually resembles \mathbf{x} as closely as possible, and (iii) maintain the integrity of the cover image \mathbf{x} using established perceptual quality metrics.

Denoising Diffusion Implicit Models (DDIMs) (Song et al. [2020]) are a class of generative models that learn the data distribution by adopting a two-phase mechanism. The forward phase incorporates noise into a clean image, while the backward phase incrementally removes the noise. The formulation for the forward diffusion in DDIM is presented as:

$$\mathbf{x}_t = \sqrt{\alpha_t} \mathbf{x}_{t-1} + \sqrt{1 - \alpha_t} \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad (1)$$

where \mathbf{x}_t is the noisy image at the t -th step, α_t is a predefined variance schedule, and t spans the discrete time steps from 1 to T . The DDIM's backward sampling equation is:

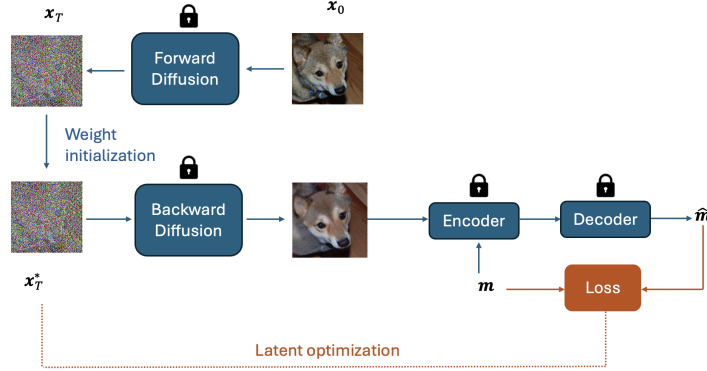


Figure 2: DDIM-based cover selection framework overview. The input cover image \mathbf{x}_0 is first converted to the latent space \mathbf{x}_T via forward diffusion. Then, guided by the message recovery loss, the latent space is fine-tuned, and the updated cover image is generated via the reverse diffusion process. The DDIM model as well as the steganographic encoder-decoder pair are pretrained.

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\mathbf{f}_\theta(\mathbf{x}_t, t) + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) + \sigma_t^2\boldsymbol{\epsilon}, \quad \mathbf{f}_\theta(\mathbf{x}_t, t) = \frac{\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)}{\sqrt{\bar{\alpha}_t}}, \quad (2)$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$, and \mathbf{f}_θ is a denoising function reliant on the pretrained noise estimator $\boldsymbol{\epsilon}_\theta$.

This sampling allows the use of different samplers by changing the variance of the noise σ_t . Especially, by setting this noise to 0, the DDIM backward process becomes deterministic, defined uniquely by the initial variable \mathbf{x}_T . This initial value can be seen as a latent code, commonly utilized in DDIM inversion, a process that utilizes DDIM to convert an image to latent noise and subsequently reconstruct it to its original form (Kim et al. [2022]).

Generative Adversarial Networks (GANs) (Goodfellow et al. [2020]) are another type of generative model designed to learn the data distribution $p(\mathbf{x})$ of a target dataset through a min-max game between two networks: a generator (G) and a discriminator (D). The generator creates synthetic samples $G(\mathbf{z})$ from a random noise vector \mathbf{z} , drawn from a simple distribution $p(\mathbf{z})$ such as a standard normal. The discriminator evaluates samples it receives—either real data \mathbf{x} from $p(\mathbf{x})$ or fake data from G —and tries to accurately classify them as real or fake. The objective of G is to generate data that D mistakes as real, while D aims to distinguish between actual and generated data effectively.

3 Methodology

We propose two cover selection methodologies using pretrained Denoising Diffusion Implicit Models (DDIM) and pretrained Generative Adversarial Networks (GAN) (Sections 3.1, 3.2), and compare the performances of the two approaches (Section 3.3). Detailed descriptions of the training procedures are in Appendix B. Broadly speaking, starting with a cover image \mathbf{x} randomly selected from the dataset, we gradually optimize this image to minimize the loss $\|\mathbf{m} - \hat{\mathbf{m}}\|$. Intriguingly, while our primary focus is on reducing the error rate, we observe that all three objectives of cover selection outlined in Section 2 are concurrently achieved. We investigate this phenomenon in Section 3.3.

3.1 DDIM-based cover selection

As depicted in Fig. 2, our DDIM approach consists of two steps. We get inspired from DDIM inversion, which refers to the process of using DDIM to achieve the conversion from an image to a latent noise and back to the original image (Kim et al. [2022]).

Step 1: latent computation. The initial cover image \mathbf{x}_0 (where the subscript denotes the diffusion step) goes through the forward diffusion process described in Eq. 3 to get the latent \mathbf{x}_T .

$$\mathbf{x}_{t+1} = \sqrt{\bar{\alpha}_{t+1}}\mathbf{f}_{\theta}(\mathbf{x}_t, t) + \sqrt{1 - \bar{\alpha}_{t+1}}\epsilon_{\theta}(\mathbf{x}_t, t) \quad (3)$$

Step 2: guided image reconstruction. We optimize \mathbf{x}_T to minimize the loss $\|\mathbf{m} - \hat{\mathbf{m}}\|$. Specifically, \mathbf{x}_T goes through the backward diffusion process described in Eq. 2 generating cover images that minimize the loss. We evaluate the gradients of the loss with respect to \mathbf{x}_T using backpropagation and use standard gradient based optimizers to get the optimal \mathbf{x}_T^* after some optimization steps.

We use a pretrained DDIM (parametrized by θ), and a pretrained LISO, the state-of-the-art steganographic encoder and decoder from Chen et al. [2022], also described in Appendix A. The weights of the DDIM and the steganographic encoder-decoder are fixed throughout \mathbf{x}_T 's optimization process.

The idea is based on the approximation of forward and backward differentials in solving ordinary differential equations (Song et al. [2020]). In the case of deterministic DDIM ($\sigma_t = 0$), Eq. 2 can be used to perform the forward and backward process (Kim et al. [2022]) and achieve accurate image reconstruction. Instead of adopting a fully deterministic DDIM, we find that having a deterministic forward process (Eq. 3) with a stochastic backward process (Eq. 2) yields better results for our setup.

3.2 GAN-based cover selection

In the GAN-based approach, we start with a latent vector \mathbf{z} randomly initialized from a Gaussian distribution, which serves as input to the generator G . The objective is to identify an optimized \mathbf{z}^* such that the cover image $G(\mathbf{z}^*)$ minimizes the loss $\|\mathbf{m} - \hat{\mathbf{m}}\|$, i.e.:

$$\mathbf{z}^* = \underset{\mathbf{z}}{\operatorname{argmin}} \|Dec(Enc(G(\mathbf{z}), \mathbf{m}) - \mathbf{m}\| \quad (4)$$

Where Enc , Dec and \mathbf{m} are the steganographic encoder, decoder and secret message respectively as described in Section 2. We evaluate the gradients of the loss with respect to \mathbf{z} using backpropagation and use standard gradient based optimizers to get the optimal \mathbf{z}^* that minimizes the loss. All other modules (Enc , Dec , G) are differentiable, pretrained and fixed during the optimization. We utilize BigGAN's pretrained conditional generator (Brock et al. [2018]), and a pretrained LISO steganographic encoder-decoder pair (Chen et al. [2022]).

Note: To achieve consistency with the DDIM approach described in Section 3.1, instead of starting with a randomly generated latent vector \mathbf{z} , we can begin a cover image \mathbf{x} and apply established GAN inversion techniques to map it to its corresponding latent space (Xia et al. [2022]).

3.3 Performance comparison: DDIM & GAN

We compare the performance of the approaches from Sections 3.1 and 3.2 in Table 1. We show the results of 10 randomly selected classes from the ImageNet dataset (Russakovsky et al. [2015]). Following Chen et al. [2022], we assess error rate defined as $\frac{\|\mathbf{m} - \hat{\mathbf{m}}\|_0}{H \times W \times B}$, the structural similarity index (SSIM) and peak signal-to-noise ratio (PSNR) (Wang et al. [2004]) to measure changes between cover and steganographic images. We further evaluate the generated cover image quality using the no-reference BRISQUE metric (Mittal et al. [2012]). Our methods outperform traditional exhaustive search techniques detailed in Section 1, which are omitted from the table for brevity.

Methods: For the DDIM-based cover selection, we generate a batch of 500 cover images, denoted as $\{\mathbf{x}_0^{(i)}\}_{i=1}^{500}$, and apply the cover selection framework to each image independently (Section 3.1). Similarly, for the GAN-based cover selection, we produce a batch of 500 randomly initialized latent vectors, represented as $\{\mathbf{z}^{(i)}\}_{i=1}^{500}$, and independently run the cover selection framework for each vector (Section 3.2). We train a steganographic encoder-decoder pair using 1000 training images from each class, adhering to the method used in Chen et al. [2022]. We then use this trained model, in addition to a diffusion model and BigGAN's conditional generator, both pretrained on ImageNet. We consider a payload of $B = 4$ bpp (we explore different payloads in Section 5.1). The secret messages are random binary bit strings, sampled from an independent $Bernoulli(0.5)$ distribution. We use the binary cross-entropy loss to optimize message recovery. For a comprehensive explanation on our hyperparameter selection, please refer to Appendix B.

Observation 1: As shown in Table 1 the optimized images produced by both DDIM and GAN exhibit significantly lower error rates compared to the original images by over **50%** for some classes.

Table 1: Comparative performance of GAN-based and DDIM-based cover selection techniques on the ImageNet dataset, with a payload $B = 4$ bpp. DDIM-optimized images achieve a significant gain over the original images and GAN-optimized images in both error rate reduction and image quality.

Classes	Error Rate (%) ↓			BRISQUE ↓			SSIM ↑			PSNR ↑		
	Original	GAN	DDIM	Original	GAN	DDIM	Original	GAN	DDIM	Original	GAN	DDIM
Robin	2.48	1.32	1.01	27.8	18.95	19.81	0.72	0.68	0.64	22.34	23.38	23.85
Snow Leopard	0.84	0.36	0.55	23.71	18.28	17.26	0.75	0.74	0.72	23.71	24.54	24.96
Daisy	1.75	0.97	1.43	9.85	9.79	7.71	0.61	0.59	0.61	26.01	26.7	26.63
Drilling Platform	2.29	1.88	1.85	25.08	25.85	27.42	0.41	0.39	0.37	21.33	21.56	21.41
Hartebeest	0.21	0.15	0.12	16.97	16.17	13.63	0.55	0.55	0.56	24.83	25.27	26.34
American Egret	0.95	0.77	0.78	24.4	22.9	12.03	0.63	0.63	0.64	22.72	22.87	24.49
Owl	0.21	0.02	0.09	26.01	27.77	21.3	0.73	0.76	0.71	24.02	24.62	26.01
Chihuahua	0.79	0.59	0.55	18.45	17.92	14.33	0.58	0.56	0.59	23.13	23.55	24.44
Cheetah	2.15	2.02	1.53	41.2	40.53	35.01	0.56	0.53	0.43	21.46	21.61	21.75
Lady's Slipper	0.17	0.08	0.07	22.53	11.13	10.24	0.71	0.68	0.76	24.65	26.13	26.15

Surprisingly, although our training objective for cover selection focused solely on minimizing the error rate, we observed improved image quality as evidenced by BRISQUE, SSIM, and PSNR scores. This intriguing relationship between higher image quality and lower error rates is further explored in Appendix H. In summary, our analysis reveals that certain image complexity metrics, including edge density and entropy, negatively correlate with both error rates and BRISQUE scores. This suggests that our cover selection framework modifies features such as edges and entropy during optimization, resulting in enhancements to both image quality and error reduction.

Observation 2: DDIM-based optimization consistently outperforms GAN-based methods across all metrics, aligning with previous findings on DDIM’s superior image generation capabilities (Dhariwal and Nichol [2021]). We further explore and compare the outputs of both methods, presenting sample steganographic images before and after optimization in Appendix G. Notably, DDIM maintains the semantic integrity of images, preserving key elements like object positions and orientations—such as a bird’s unchanged gaze. In contrast, GANs may significantly modify an image’s composition, even altering a bird’s gaze from left to right, which impacts its semantic content.

For the remainder of the paper, we will utilize the DDIM-based approach, due to its enhanced performance in both error reduction and image quality.

4 Analysis

In this section, we explore the reasons behind the enhanced performance achieved by our framework. Initially, we analyze the behavior of the pretrained steganographic encoder (Section 4.1). Our observations indicate that the encoder preferentially embeds messages within pixels of low variance. To validate these findings, we compare the encoder’s behavior with the waterfilling technique applied to parallel Gaussian channels (Section 4.2). Lastly, we demonstrate that the cover selection optimization effectively increases the presence of low variance pixels. This adjustment equips the encoder with greater flexibility to hide messages, thereby improving overall performance (Section 4.3). We present the results for the ImageNet Robin class with a payload of $B = 4$ bpp. Additional results for various classes and datasets are presented in Appendix D.

4.1 Encoding in low-variance pixels

We begin by investigating the underlying mechanism of the pretrained steganographic encoder (Chen et al. [2022]). We hypothesize that the encoder preferentially hides messages in regions of low pixel variance. To test this hypothesis, we structure our analysis into two steps.

Step 1: variance analysis. In Fig. 3 (top), we illustrate the variance of each pixel position for the three color channels, calculated across a batch of images and normalized to a range between 0 and 1, as detailed in Appendix D. The plot reveals significant disparities in variance, with certain regions displaying notably lower variance compared to others.

Step 2: residual computation. Using the same batch of images, we pass them through the steganographic encoder to obtain the corresponding steganographic images. We then compute the residuals by calculating the absolute difference between the cover and steganographic images and averaging

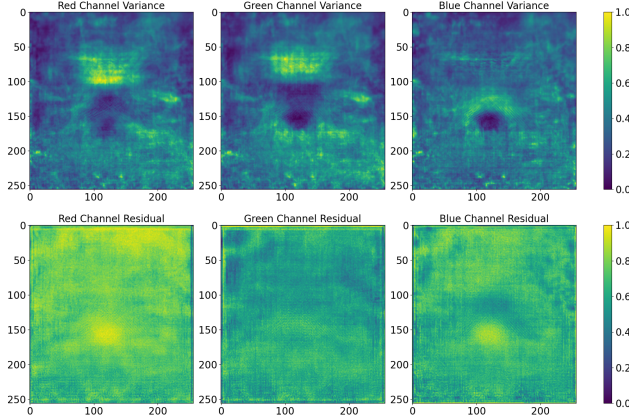


Figure 3: Normalized pixel variances (top) and residuals (bottom) calculated across a batch of 500 Robin images for each color channel, before optimization.

these differences across the batch. This process yields three maps, one for each color channel, which are subsequently normalized to a range between 0 and 1. Those maps are plotted in Fig. 3 (bottom).

As shown in Fig. 3, we observe correlations between the variance and the magnitude of the residual values; where pixels with lower-variance tends to have higher residual magnitudes. To quantify this observation, we introduced a threshold value of 0.5. In the residual maps (from Step 2), locations exceeding this threshold are classified as “high-message regions” and assigned a value of 1. Conversely, locations in the variance maps (from Step 1) falling below this threshold are defined as “low-variance regions”, also set to 1. We discovered that **81.6%** of the high-message regions coincide with low-variance pixels. This substantial overlap confirms our hypothesis and underscores the encoder’s tactic of utilizing low-variance areas to embed messages, which is a highly desired and natural behavior. We highlight that we are the first to make this observation, despite there being several relevant works on learning-driven steganography; none of these prior studies conducted an interpretation analysis of the encoder to uncover this behavior.

Interestingly, we find that the learned message embedding behavior closely aligns with the waterfilling strategy, the theoretically optimal embedding strategy for parallel Additive Gaussian Noise channels, a fundamental concept in communication theory (Cover [1999]). This strategy involves embedding more messages in lower-variance pixel positions, which increases message recovery accuracy. Surprisingly, steganography methods tend to adopt this strategy implicitly, without explicit training to do so. In the subsequent section, we delve deeper into this analogy and further demonstrate the relationship between these two processes.

4.2 Analogy to waterfilling

To validate the findings presented in Section 4.1, we draw parallels between our analysis and the waterfilling problem for Gaussian channels. We conceptualize the process of hiding secret messages as transmitting information through N parallel communication channels, where N corresponds to the number of pixels in an image. In this analogy, each pixel operates as an individual communication link, with the secret message functioning as the signal to be hidden and later recovered. The cover image, which embeds the hidden message, serves as noise unknown to the decoder.

We consider a simple additive steganography scheme: $s_i = x_i + \gamma_i m_i$, for $i = 1, 2, \dots, N$, where $N = H \times W \times 3$ is the image dimension, $m_i = \{-1, 1\}$ indicates the i -th message to be embedded, γ_i its corresponding power, x_i and s_i represent the i -th element of the cover and steganographic images respectively. We assume a power constraint P that restricts the deviation between the cover and steganographic images: $E \left[\sum_{i=1}^N (s_i - x_i)^2 \right] \leq P$.

This formulation is similar to the waterfilling solution for N parallel Gaussian channels (Cover [1999]), where the objective is to distribute the total power P among the N channels so as to maximize the capacity C , which is maximum rate at which information can be reliably transmitted

over a channel, defined as: $C = \sum_{i=1}^N \log_2 \left(1 + \frac{\gamma_i^2}{\sigma_i^2} \right)$, where σ_i^2 is the variance of x_i . The problem can be formulated as a constrained optimization problem, where the optimal power allocation is given by $\gamma_i^2 = \left(\frac{1}{\lambda \ln(2)} - \sigma_i^2 \right)^+$, where $(x)^+ = \max(x, 0)$ and λ is chosen to satisfy the power constraint.

We calculate $\{\sigma_i^2\}_{i=1}^{3 \times H \times W}$ using a batch of images, and find the optimized $\{\gamma_i^2\}_{i=1}^{3 \times H \times W}$ using the approach described above. We plot the γ_i 's for each color channel in Fig. 4.

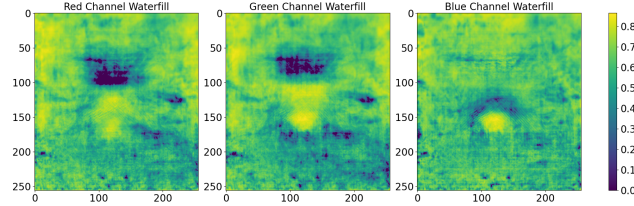


Figure 4: Power coefficients γ_i for each color channel, calculated using a batch of 500 Robin images.

We observe a degree of similarity when comparing with Fig. 3 (bottom). To quantitatively assess this resemblance across color channels, we quantize the three matrices by setting values greater than 0.5 to 1 and values less than 0.5 to 0. For each channel, the similarity is calculated using the equation $\frac{\sum_{i,j} \mathbf{1}(\mathbf{W}_{ij}^{(k)} = \mathbf{R}_{ij}^{(k)})}{256 \times 256}$, where $\mathbf{W}_{ij}^{(k)}$ and $\mathbf{R}_{ij}^{(k)}$ are the (i, j) -th pixels of the quantized waterfilling and residual matrices, respectively, for the channel k . The computed similarity scores are **81.8%** for red, **65.5%** for green, and **74.9%** for blue, revealing varying degrees of resemblance with the waterfilling strategy across the color channels. The variation underscores that the waterfilling strategy is implemented more effectively in some channels than in others.

4.3 Impact of cover selection

A natural question becomes: what is the cover selection optimization doing? We plot the variance maps of the optimized cover images in Fig. 5.

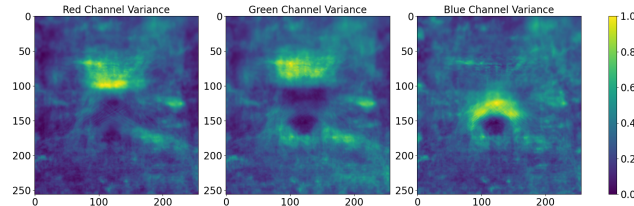


Figure 5: Normalized pixel variances across a batch of 500 Robin images for each color channel, after optimization.

We notice that the number of low variance spots significantly increased as compared to Fig. 3 (top), meaning that the encoder has more freedom in encoding the secret message. Quantitatively, we find that **92.4%** of the identified high-message positions are encoded in low-variance pixels, as compared to **81.6%** before optimization. Given that the encoder preferentially embeds data in these low variance areas, this increase provides greater flexibility for data embedding, thereby explaining the performance gains observed in our framework.

5 Practical settings

In this section, we adapt our framework for practical considerations. We evaluate its performance across different payloads (Section 5.1), adapt it for JPEG compression (Section 5.2), and confirm security against steganalysis (Section 5.3). Computational times are detailed in Appendix I. We use two datasets, CelebA-HQ (Karras et al. [2017]) and AFHQ-Dog (Choi et al. [2020]), using the same settings described in Section 3.3.

Table 2: Performance comparison across AFHQ-Dog and CelebA-HQ across various payloads. We observe that the error rates of DDIM-optimized images are significantly lower than original images.

	Payload B	Error Rate (%) ↓		BRISQUE ↓		SSIM ↑		PSNR ↑	
		Original	DDIM	Original	DDIM	Original	DDIM	Original	DDIM
CelebA-HQ	1 bpp	2.6E-04	1.5E-05	2.75	4.07	0.95	0.94	36.25	36.37
	2 bpp	2.3E-03	9E-04	5.9	9.7	0.91	0.92	31.82	32.46
	3 bpp	0.011	0.002	9.95	9.83	0.86	0.87	32.16	33.88
	4 bpp	0.051	0.019	11.91	11.04	0.81	0.83	30.91	32.46
AFHQ-Dog	1 bpp	8E-05	0.00	12.14	12.11	0.94	0.93	36.84	36.87
	2 bpp	8E-04	6.8E-05	4.12	7.19	0.93	0.94	35.1	34.4
	3 bpp	0.007	0.002	10.34	6.87	0.86	0.85	32.5	32.6
	4 bpp	0.11	0.09	13.49	13.42	0.75	0.76	28.97	28.99

5.1 Payload impact on performance

We explore different payload capacities B , highlighted in Table 2. We show the results for $B = 1, 2, 3, 4$ bits per pixel (bpp). DDIM-optimized images show error rates significantly lower than originals, with image quality metrics like BRISQUE, SSIM, and PSNR largely preserved, though some quality decline was noted at lower bpp levels in CelebA-HQ and AFHQ-Dog. We include sample generated cover images generated using the DDIM framework in Appendix E. Despite experimenting with various regularization techniques aimed at maintaining image quality, no noticeable improvement was observed (Appendix C). Considering this, extending our framework to explore novel regularization techniques for such payload capacities is an interesting future direction. We also provide example cover and steganographic images generated by the LISO framework under different payload values in Appendix F.

5.2 JPEG compression

Robustness against lossy image compression is crucial for steganography. We extend our framework to accommodate JPEG compression (Wallace [1991]). Following Athalye et al. [2018], we implement an approximate JPEG layer where the forward pass executes standard JPEG compression, while the backward pass operates as an identity function. Once the encoder-decoder pair is trained, we generate a JPEG-compliant cover image following the framework described in Section 3.1, augmented by adding a JPEG layer post-encoding. In Table 3, we demonstrate that our framework achieves improved error rates for $B = 1$ bpp, thereby validating our approach’s capability to optimize cover images under JPEG compression constraints. In addition, we show robustness results to Gaussian noise in Appendix K.

Table 3: JPEG results for $B = 1$ bpp.

Dataset	Error Rate % ↓		PSNR ↑	
	Original	DDIM	Original	DDIM
CelebA-HQ	0.12	0.06	21.09	21.53
AFHQ-Dog	0.15	0.11	19.34	19.63

Table 4: Steganalysis results AFHQ-Dog.

	Payload B	Error Rate (%) ↓		XuNet Det. (%) ↓	
		Original	DDIM	Original	DDIM
Scenario 1	1 bpp	8E-05	0.00	37.1	37.5
	2 bpp	8E-04	6.8E-05	31.34	15.42
	3 bpp	0.007	0.002	20.39	34.82
	4 bpp	0.11	0.09	97.37	97.35
Scenario 2	1 bpp	0.0026	2E-05	0.0	0.0
	2 bpp	0.0024	1E-04	0.0	0.0
	3 bpp	0.01	0.003	3.2	2.1
	4 bpp	0.23	0.22	9.2	8.6

5.3 Steganalysis

Steganalysis systems are designed to detect whether there is hidden information within images. As these tools evolve, neural steganography techniques now integrate these systems into their end-to-end pipelines to create images that can bypass detection (Chen et al. [2022], Shang et al. [2020]). We show our results in Table 4 on the AFHQ-Dog dataset. Following the approach in Chen et al. [2022], we evaluate the security of our optimized images by measuring the detection rate using the steganalysis tool XuNet (Xu et al. [2016]) and also record message recovery error rates. The image quality metrics, such as BRISQUE, SSIM, and PSNR, are comparable to those listed in Table 2 and have therefore been omitted for brevity. We explore two different scenarios:

Scenario 1: In this scenario, the experimental setup remains the same as described in Section 3.1 and illustrated in Fig. 2. The steganographic encoder-decoder pair is trained without regularizers to evade steganalysis detection. The DDIM-optimized images exhibit comparable detection rates at payloads of $B = 1$ and $B = 4$, superior performance at $B = 2$, and inferior performance at $B = 3$, all while achieving significantly lower error rates. While it is puzzling that detection rates do not consistently decrease with lower payload size, this phenomenon is also observed in LISO Chen et al. [2022], on which our framework is built. We provide a more detailed discussion in Appendix J.

Scenario 2: We leverage the differentiability of XuNet as described in Chen et al. [2022]. During the optimization of the steganographic encoder-decoder pair, we introduce an additional loss term to account for steganalysis. This adjustment leads to a notable reduction in detection rates across all payload sizes, while maintaining consistently low error rates for both original and DDIM-optimized images. Notably, DDIM-optimized images exhibit even lower detection and error rates compared to the original images, demonstrating superior performance.

Further implementation details, along with results using an alternative steganalysis method, SRNet (Boroumand et al. [2018]), are provided in Appendix J.

6 Conclusion

We propose a novel cover selection framework for steganography leveraging pretrained generative models. We demonstrate that by carefully optimizing the latent space of these models, we generate steganographic images that exhibit high visual quality and embedding capacity. Additionally, our information-theoretic analysis shows that message hiding predominantly occurs in low-variance pixels, reflecting the waterfilling algorithm’s approach to parallel Gaussian channels. Our framework is versatile, allowing for the incorporation of further constraints to produce JPEG-resistant steganographic images or to evade detection by particular steganalysis systems. For future work, we aim to expand our analysis (Section 4.2) to draw similarities with correlated Gaussian channels, moving beyond the independent channels considered in this work.

Acknowledgments

This work was partly supported by ARO Award W911NF2310062, ONR Award N000142412542, and the 6G@UT center within the Wireless Networking and Communications Group (WNCG) at the University of Texas at Austin.

References

- Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International conference on machine learning*, pages 274–283. PMLR, 2018.
- Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18208–18218, 2022.
- Shumeet Baluja. Hiding images in plain sight: Deep steganography. *Advances in neural information processing systems*, 30, 2017.
- Ifra Bilal, Rajiv Kumar, Mahendra Singh Roj, and PK Mishra. Recent advancement in audio steganography. In *2014 International Conference on Parallel, Distributed and Grid Computing*, pages 402–405. IEEE, 2014.
- Mehdi Boroumand, Mo Chen, and Jessica Fridrich. Deep residual network for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 14(5):1181–1193, 2018.
- Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018.

- Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023.
- Tu Bui, Shruti Agarwal, Ning Yu, and John Collomosse. Rosteals: Robust steganography using autoencoder latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 933–942, 2023.
- Abbas Cheddad, Joan Condell, Kevin Curran, and Paul Mc Kevitt. Digital image steganography: Survey and analysis of current methods. *Signal processing*, 90(3):727–752, 2010.
- Xiangyu Chen, Varsha Kishore, and Kilian Q Weinberger. Learning iterative neural optimizers for image steganography. In *The Eleventh International Conference on Learning Representations*, 2022.
- Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8188–8197, 2020.
- Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.
- Ingemar Cox, Matthew Miller, Jeffrey Bloom, Jessica Fridrich, and Ton Kalker. *Digital watermarking and steganography*. Morgan kaufmann, 2007.
- Ingemar J Cox, Matthew L Miller, and Andrew L McKellips. Watermarking as communications with side information. *Proceedings of the IEEE*, 87(7):1127–1141, 1999.
- Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.
- Oleg Evsutin, Anna Kokurina, and Roman Meshcheryakov. Approach to the selection of the best cover image for information embedding in jpeg images based on the principles of the optimality. *Journal of Decision Systems*, 27(sup1):256–264, 2018.
- Jessica Fridrich, Miroslav Goljan, and Rui Du. Detecting lsb steganography in color, and gray-scale images. *IEEE multimedia*, 8(4):22–28, 2001.
- Oran Gafni, Adam Polyak, Oron Ashual, Shelly Sheynin, Devi Parikh, and Yaniv Taigman. Make-a-scene: Scene-based text-to-image generation with human priors. In *European Conference on Computer Vision*, pages 89–106. Springer, 2022.
- Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.
- Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.
- Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.
- Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.
- Mehdi Kharrazi, Husrev T Sencar, and Nasir Memon. Cover selection for steganographic embedding. In *2006 International Conference on Image Processing*, pages 117–120. IEEE, 2006.
- Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2426–2435, 2022.
- Varsha Kishore, Xiangyu Chen, Yan Wang, Boyi Li, and Kilian Q Weinberger. Fixed neural network steganography: Train the images, not the network. In *International Conference on Learning Representations*, 2021.

- Anish Mittal, Anush Krishna Moorthy, and Alan Conrad Bovik. No-reference image quality assessment in the spatial domain. *IEEE Transactions on image processing*, 21(12):4695–4708, 2012.
- Pierre Moulin and Joseph A O’Sullivan. Information-theoretic analysis of information hiding. *IEEE Transactions on information theory*, 49(3):563–593, 2003.
- Pierre Moulin, Mehmet Kivanç Mihcak, and Gen-Iu Lin. An information-theoretic model for image watermarking and data hiding. In *Proceedings 2000 International Conference on Image Processing (Cat. No. 00CH37101)*, volume 3, pages 667–670. IEEE, 2000.
- Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.
- Jiaohua Qin, Yuanjing Luo, Xuyu Xiang, Yun Tan, and Huajun Huang. Coverless image steganography: a survey. *IEEE access*, 7:171372–171394, 2019.
- Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.
- Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.
- Yueyun Shang, Shunzhi Jiang, Dengpan Ye, and Jiaqing Huang. Enhancing the security of deep learning steganography via adversarial examples. *Mathematics*, 8(9):1446, 2020.
- Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9243–9252, 2020.
- Haichao Shi, Jing Dong, Wei Wang, Yinlong Qian, and Xiaoyu Zhang. Ssgan: Secure steganography based on generative adversarial networks. In *Advances in Multimedia Information Processing—PCM 2017: 18th Pacific-Rim Conference on Multimedia, Harbin, China, September 28-29, 2017, Revised Selected Papers, Part I 18*, pages 534–544. Springer, 2018.
- Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020.
- Yeshwanth Srinivasan, Brian Nutter, Sunanda Mitra, Benny Phillips, and Daron Ferris. Secure transmission of medical records using high capacity steganography. In *Proceedings. 17th IEEE Symposium on Computer-Based Medical Systems*, pages 122–127. IEEE, 2004.
- Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
- Matthew Tancik, Ben Mildenhall, and Ren Ng. Stegastamp: Invisible hyperlinks in physical photographs. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2117–2126, 2020.
- Gregory K Wallace. The jpeg still picture compression standard. *Communications of the ACM*, 34(4):30–44, 1991.
- Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.
- Zichi Wang and Xinpeng Zhang. Secure cover selection for steganography. *IEEE Access*, 7:57857–57867, 2019.
- Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. Gan inversion: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(3):3121–3138, 2022.

- Guanshuo Xu, Han-Zhou Wu, and Yun-Qing Shi. Structural design of convolutional neural networks for steganalysis. *IEEE Signal Processing Letters*, 23(5):708–712, 2016.
- Farzin Yaghmaee and Mansour Jamzad. Estimating watermarking capacity in gray scale images based on image complexity. *EURASIP Journal on Advances in Signal Processing*, 2010:1–9, 2010.
- Jiwen Yu, Xuanyu Zhang, Youmin Xu, and Jian Zhang. Cross: Diffusion model makes controllable, robust and secure image steganography. *Advances in Neural Information Processing Systems*, 36, 2024.
- Kevin Alex Zhang, Alfredo Cuesta-Infante, Lei Xu, and Kalyan Veeramachaneni. Steganogan: High capacity image steganography with gans. *arXiv preprint arXiv:1901.03892*, 2019.
- Zhili Zhou, Huiyu Sun, Rohan Harit, Xianyi Chen, and Xingming Sun. Coverless image steganography without embedding. In *Cloud Computing and Security: First International Conference, ICCCS 2015, Nanjing, China, August 13-15, 2015. Revised Selected Papers 1*, pages 123–132. Springer, 2015.

A Learned Iterative Steganography Optimization (LISO)

LISO (Chen et al. [2022]) advances the method established in Kishore et al. [2021], which is centered around Optimization-based Image Steganography. Leveraging a differentiable decoder equipped with either randomly initialized or pretrained weights (as referenced in the preceding paragraph), Kishore et al. [2021] formulates the steganography encoding as an optimization task for each sample. This approach is similar to the generation of adversarial perturbations as discussed in Szegedy et al. [2013]. Specifically, the technique described in Kishore et al. [2021] seeks to compute a steganographic image by addressing a constrained optimization problem that ensures the perturbed image remains within the bounds of the $[0, 1]^{H \times W \times 3}$ hypercube.

$$\min_{\mathbf{s} \in [0,1]^{H \times W \times 3}} L_{\text{acc}}(\text{Dec}(\mathbf{s}), \mathbf{m}) + \lambda L_{\text{qua}}(\mathbf{s}, \mathbf{x}) \quad (5)$$

where

$$L_{\text{acc}}(\hat{\mathbf{m}}, \mathbf{m}) := \langle \mathbf{m}, \log \hat{\mathbf{m}} \rangle + \langle (1 - \mathbf{m}), \log(1 - \hat{\mathbf{m}}) \rangle \quad (6)$$

$$L_{\text{qua}}(\mathbf{s}, \mathbf{x}) := \frac{1}{N} \|\mathbf{s} - \mathbf{x}\|^2, \quad (7)$$

where \mathbf{m} represents the secret message, $\hat{\mathbf{m}}$ is the decoded message, \mathbf{x} is the cover image, and \mathbf{s} is the steganographic image. The operation $\langle \cdot \rangle$ signifies the dot product, λ is a scaling factor, and $N = H \times W \times 3$ represents the total number of pixels in the image, with H and W being the height and width of the image, respectively. The accuracy loss, $L_{\text{acc}}(\hat{\mathbf{m}}, \mathbf{m})$, is calculated using binary cross entropy to minimize the distance between the estimated and actual messages, while the quality loss, $L_{\text{qua}}(\mathbf{s}, \mathbf{x})$, employs mean squared error to ensure the steganographic image closely resembles the cover image. This objective function is represented as $\ell(\mathbf{x}, \mathbf{m})$. To solve the optimization problem outlined above, various solvers can be utilized and as shown in Algorithm 1, with iterative, gradient-based algorithms. In Algorithm 1, $\eta > 0$ is the step size, and $g(\cdot)$ describes the update function specific to the optimization method used. The perturbation δ is iteratively adjusted to minimize the loss ℓ while adhering to the pixel constraints of the image.

Algorithm 1 Iterative Optimization

```

1:  $\delta_0 \leftarrow 0$ 
2: for  $t = 1$  to  $T$  do
3:    $\delta_t \leftarrow \delta_{t-1} + \eta \cdot g(\nabla_{\delta} \ell(\mathbf{x} + \delta_{t-1}, \mathbf{m}), \mathbf{x}, \delta_{t-1})$ 
4: end for
5:  $\mathbf{s} \leftarrow \mathbf{x} + \delta_T$ 

```

In LISO, The function $g(\cdot)$ in Algorithm 1 is approximated using a fully convolutional network designed around a gated recurrent unit (GRU). The complete LISO framework, which includes the iterative encoder, decoder, and critic, undergoes end-to-end training on a diverse image dataset. Similar to the training process of Generative Adversarial Networks (GANs), the training of LISO alternates between optimizing the critic and the encoder-decoder networks. Throughout this training phase, losses for all intermediate updates are calculated with exponentially increasing weights (γ^{T-t} at step t). With intermediate predictions denoted as $\hat{\mathbf{m}}_1, \dots, \hat{\mathbf{m}}_T$ the loss is:

$$L_{\text{train}} = \sum_{t=1}^T \gamma^{T-t} [L_{\text{acc}}(\mathbf{m}, \hat{\mathbf{m}}_t) + \lambda L_{\text{qua}}(\mathbf{x}, \mathbf{s}_t) + \mu L_{\text{crit}}(\mathbf{x}, \mathbf{s}_t)],$$

where $\gamma \in (0, 1)$ is a decay factor and L_{crit} denotes the critic loss to generate real-looking images (with weight $\mu > 0$).

B Training details

B.1 GAN-based cover selection

In our GAN-based cover selection method, we utilize the BigGAN generator (Brock et al. [2018]) and a LISO encoder-decoder pair (Chen et al. [2022]), both pretrained on the ImageNet dataset

(Russakovsky et al. [2015]). Specifically, the BigGAN generator receives a latent vector \mathbf{z} , a 128-dimensional vector initialized from a truncated normal distribution with truncation set at 0.4, and a class index c . It then produces the cover image $\mathbf{x} \in [0, 1]^{H \times W \times 3}$. The LISO encoder processes \mathbf{x} along with the secret message $\mathbf{m} \in \{0, 1\}^{H \times W \times B}$ to create the steganographic image \mathbf{s} , while the LISO decoder attempts to recover $\hat{\mathbf{m}}$ from \mathbf{s} . We consider a payload $B = 4$. To optimize the latent vector \mathbf{z} , we minimize the binary cross-entropy loss $BCE(\mathbf{m}, \hat{\mathbf{m}})$ using the Adam optimizer with a learning rate of 0.01 over 100 epochs. Both the GAN generator and the LISO encoder-decoder are configured with the same architecture and parameters as described in their respective original publications.

To replicate the results presented in Table 1, we optimize a batch of 500 latent vectors $\{\mathbf{z}^{(i)}\}_{i=1}^{500}$ for each class. These vectors are randomly initialized and subsequently optimized. We then report several metrics: the average error rate between the original message \mathbf{m} and the estimated message $\hat{\mathbf{m}}$, the average BRISQUE scores of the cover images to assess their naturalness, and both the SSIM (Structural Similarity Index) and PSNR (Peak Signal-to-Noise Ratio) values to evaluate the similarity and quality between the cover and steganographic image pairs.

B.2 DDIM-based cover selection

In our cover selection method based on Denoising Diffusion Implicit Models (DDIM), we employ three DDIM models alongside LISO encoder-decoder pairs, each pretrained on different datasets: ImageNet (Russakovsky et al. [2015]), AFHQ-Dog (Choi et al. [2020]), and CelebA-HQ (Karras et al. [2017]).

Following the procedure outlined in Section 3.1, we initiate the process by sampling a random image \mathbf{x}_0 . We then execute the deterministic forward DDIM process over T steps, with each step defined as follows:

$$\mathbf{x}_{t+1} = \sqrt{\bar{\alpha}_{t+1}} \mathbf{f}_\theta(\mathbf{x}_t, t) + \sqrt{1 - \bar{\alpha}_{t+1}} \epsilon_\theta(\mathbf{x}_t, t) \quad (8)$$

After obtaining the latent representation \mathbf{x}_T , we initiate the stochastic reverse DDIM process, which spans E epochs. Within each epoch, we perform the reverse DDIM process on the acquired latent for N iterations. Each iteration proceeds as follows:

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \mathbf{f}_\theta(\mathbf{x}_t, t) + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} \epsilon_\theta(\mathbf{x}_t, t) + \sigma_t^2 \epsilon \quad (9)$$

Where ϵ_θ is a pretrained network, \mathbf{f}_θ is a function of ϵ_θ , $\sigma_t^2 = \sqrt{0.5 \cdot \left(\left(1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}} \right) \cdot \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \right)}$, and $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

We configure our model with the following parameters: $E = 50$ epochs, $T = 40$ time steps, and $N = 6$ iterations per epoch. For optimization, we employ the Adam optimizer with a learning rate of $2E - 06$. The variance schedule that determines $\bar{\alpha}_t$ and $\bar{\alpha}_{t-1}$, as well as the DDIM architectures, are consistent with those described in Kim et al. [2022].

C Regularization effect

Despite testing several regularization methods intended to preserve image quality—including total variation (Rudin et al. [1992]), edge preservation (Perona and Malik [1990]), feature matching with a pre-trained VGG network (Gatys et al. [2015]), and a classic l_1 distance between the original and updated cover images—we observed no significant enhancements. These results are shown in Table 5.

D Encoding operation analysis: additional results

In this section, we further describe the encoder’s strategy of embedding messages in regions with low pixel variance, as described in Section 4.1.

Table 5: Performance results with regularization on CelebA-HQ with a payload $B = 2$ bpp. We show the results for edge preservation (EP), l_1 distance between original and updated cover images (l_1), total variation (TV), and VGG feature matching (VGG).

Method	Error Rate (%) ↓		BRISQUE ↓		SSIM ↑		PSNR ↑	
	Original	DDIM	Original	DDIM	Original	DDIM	Original	DDIM
EP	2E-03	7E-04	11.62	13.25	25.57	25.65	0.85	0.85
l_1	2E-03	1E-03	11.94	13.45	25.65	25.7	0.85	0.85
TV	2E-03	7E-04	11.58	13.43	25.48	25.57	0.85	0.85
VGG	2.1E-03	7.5E-04	11.32	13.65	25.59	25.65	0.85	0.85

We calculate the variance for each pixel position across a batch of 500 images, separately for each of the three color channels. This results in three variance maps, each of shape 256x256 (corresponding to the dimensions of the images). These variance maps are normalized to a range between 0 and 1 to facilitate subsequent analysis.

We present variance and residual maps for three additional ImageNet classes: Daisy (Fig. 6), Yellow Lady’s Slipper (Fig. 7), and American Egret (Fig. 8). These visualizations validate that the encoder predominantly conceals messages within areas of low variance. Further analysis includes the CelebA-HQ dataset with a payload of $B = 2$ bpp, illustrated in Fig. 9. Notably, in the Blue channel, the encoder distinctly favors low variance pixels for message concealment. Intriguingly, in the Green channel, regions such as the eyes, nose, and mouth are preferred for embedding messages. Investigating the underlying reasons for this selective use is an interesting open problem.

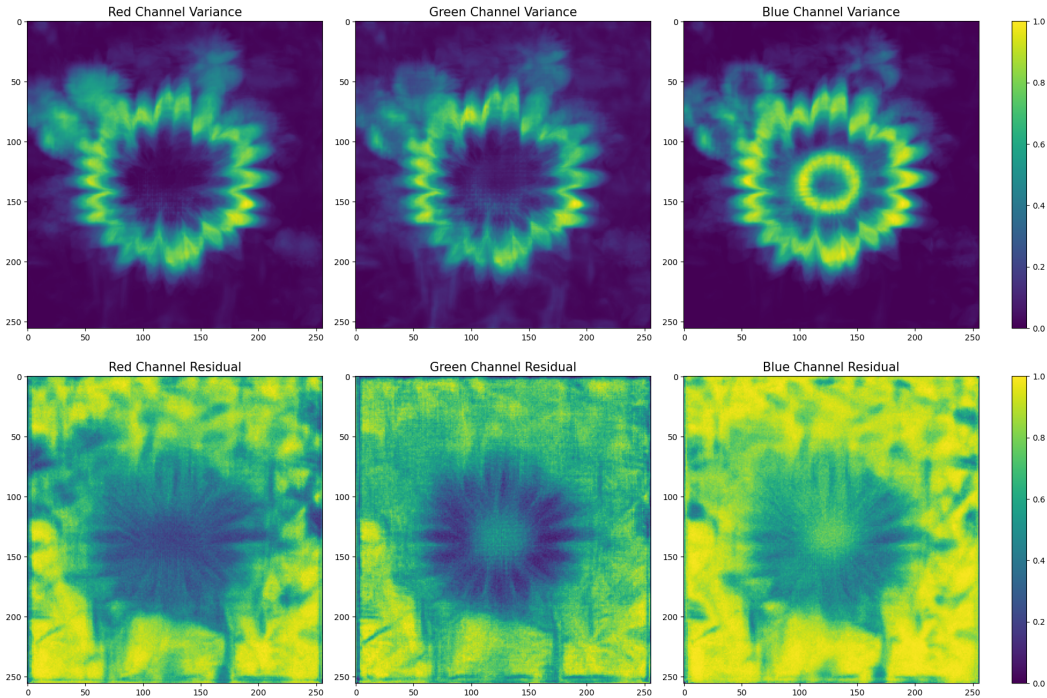


Figure 6: Normalized pixel variances (top) and residuals (bottom) across a batch of 500 images for the ImageNet Daisy class.

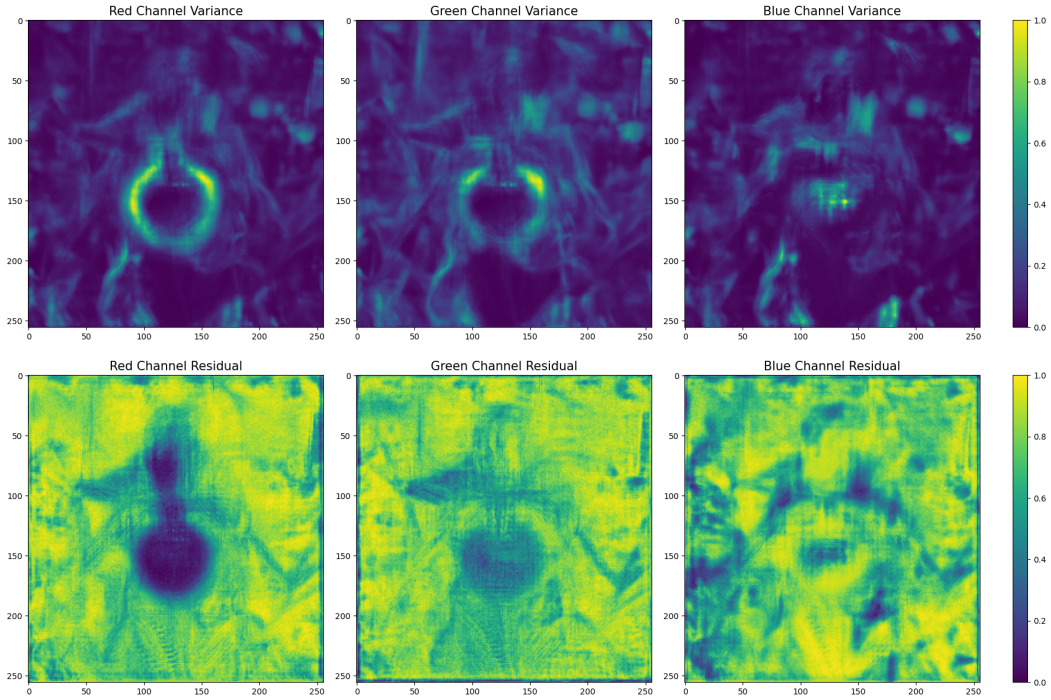


Figure 7: Normalized pixel variances (top) and residuals (bottom) across a batch of 500 images for the ImageNet Yellow Lady's Slipper class.

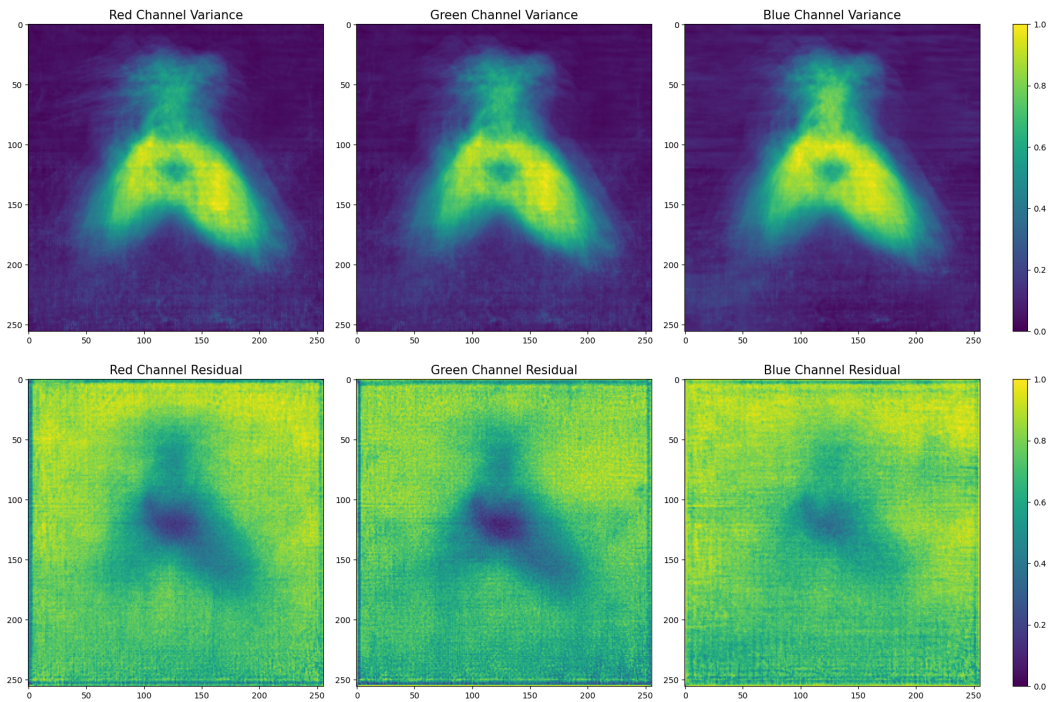


Figure 8: Normalized pixel variances (top) and residuals (bottom) across a batch of 500 images for the ImageNet American Egret class.

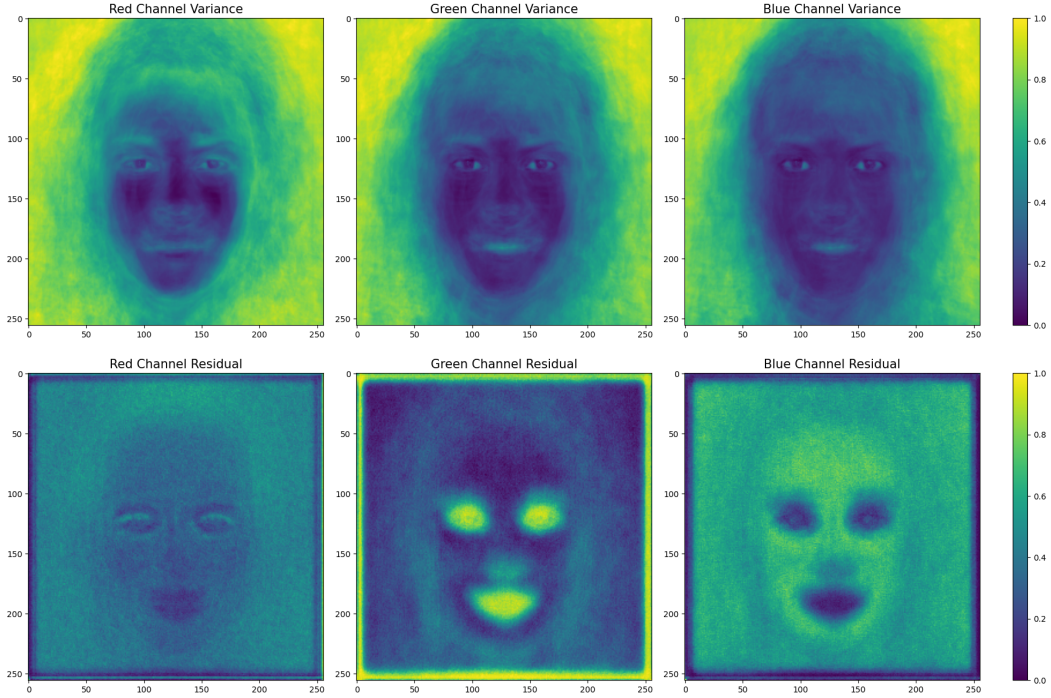


Figure 9: Normalized pixel variances (top) and residuals (bottom) across a batch of 500 images for the CelebA-HQ dataset.

E DDIM sample cover images

In this section, we present optimized cover images generated by our DDIM cover selection framework. Samples from both the CelebA-HQ and AFHQ-Dog datasets are displayed, showcasing variations for different payload capacities with $B = 1, 2, 3, 4$ bits per pixel (bpp).

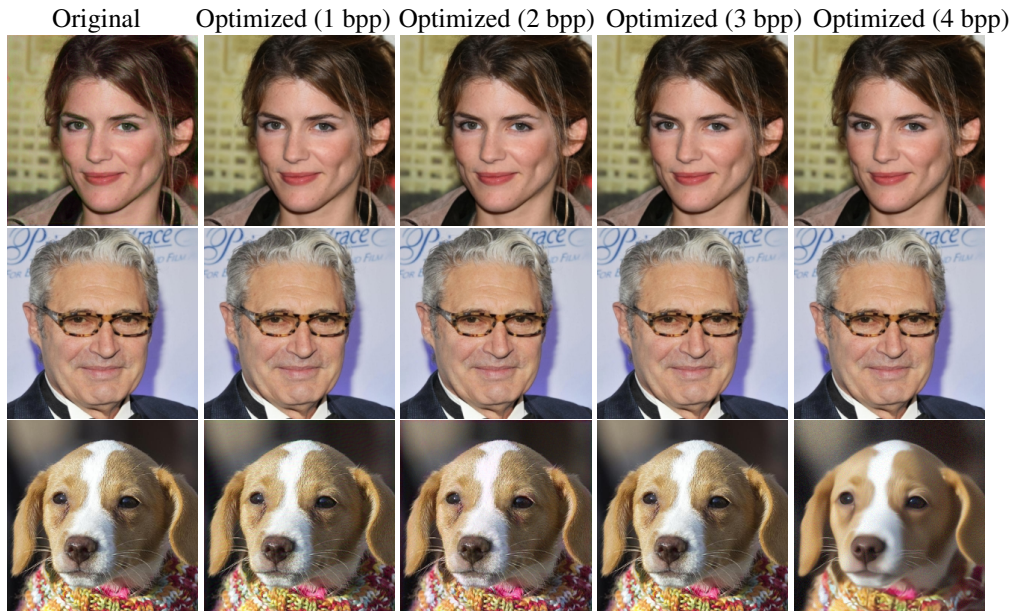


Figure 10: Generated DDIM cover images for different message payload values.

F Sample steganographic images

In this section, we present a selection of randomly sampled cover images from CelebA-HQ and AFHQ alongside their steganographic counterparts generated using the LISO framework (Chen et al. [2022]). The results are demonstrated for various payload capacities, ranging from $B = 1$ to 4 bits per pixel (bpp).



Figure 11: Covers and their corresponding steganographic images.

G Sample steganographic images: DDIM vs GAN

We compare the outputs of both methods, presenting sample steganographic images before and after optimization in Fig 12 for a payload $B = 4$ bits per pixel. DDIM conserves the semantic essence of images, maintaining critical aspects such as the positions and orientations of objects—for instance, a bird’s gaze remains consistent. In contrast, GANs can substantially alter an image’s structure, potentially changing a bird’s gaze direction, thus affecting its semantic meaning.

H Image complexity metrics

In this section, we explore the intriguing observation that optimizing for error rate not only preserves image quality but, in some instances, even enhances it. This occurs despite the fact that our primary focus is not directly on image quality optimization.

We assess various complexity metrics—entropy, edge density, compression ratio, and color diversity—across a dataset of 500 images from the AFHQ-Dog collection, each embedded with a payload of $B = 4$ bits per pixel. Our analysis investigates how these metrics correlate with the message error rate, as depicted in Figure 13. Furthermore, we investigate the relationship between these complexity metrics and the BRISQUE image quality score, as shown in Figure 14.

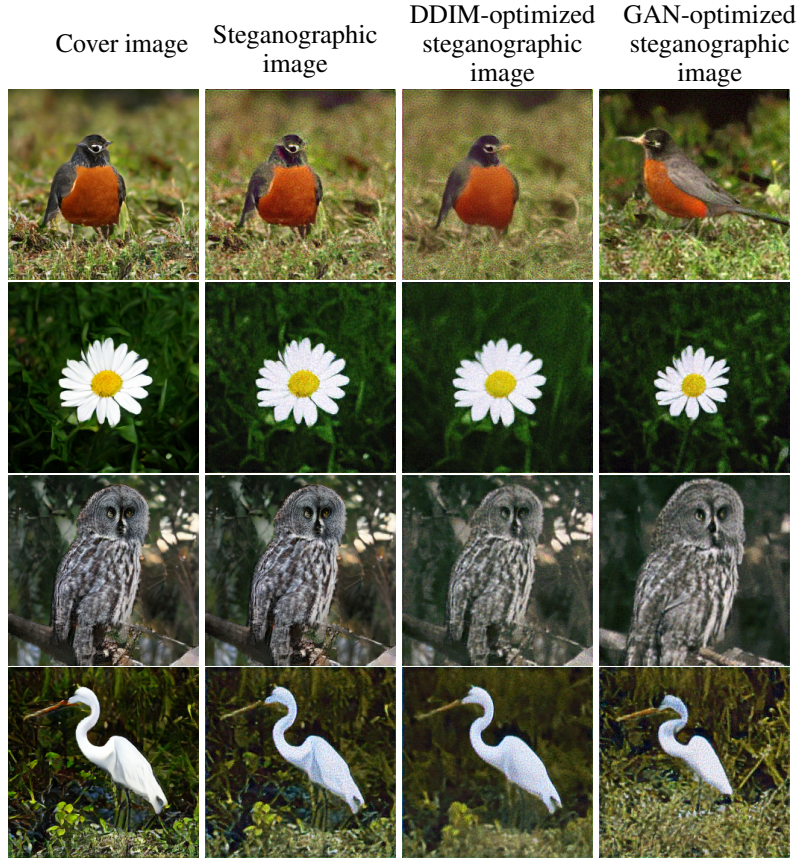


Figure 12: Generated steganographic images: GAN vs DDIM.

The entropy of an image measures the randomness of intensity values and is calculated as $-\sum_{i=1}^{256} p_i \log_2 p_i$, where p_i is the probability of occurrence of the i^{th} intensity value, calculated over a batch of images. The probabilities are determined from the grayscale version of each image C , where the grayscale conversion simplifies the entropy calculation by focusing on the luminance information while discarding color details.

The edge density of an image measures the proportion of pixels that are part of edges to the total number of pixels in the image. This is typically calculated by first applying an edge detection algorithm, such as the Sobel or Canny operator, to identify edge pixels. The edge density is then quantified as $\frac{n_e}{N}$, where n_e is the number of edge pixels identified, and N is the total number of pixels in the image.

The compression ratio of an image is a measure of the efficiency of a compression algorithm, defined as the ratio of the original file size to the compressed file size. Mathematically, it can be expressed as $\frac{S_{original}}{S_{compressed}}$ where $S_{original}$ is the original file size in bytes, and $S_{compressed}$ is the size of the file after compression. A higher compression ratio indicates more effective compression, reducing storage and transmission resource requirements.

The color diversity in an image refers to the variety and distribution of colors present within the image. It can be quantified by analyzing the image's color histogram, which represents the frequency of each color in the image. Color diversity is often measured using metrics such as the number of distinct colors, or the evenness of their distribution. A common approach is to calculate the Shannon diversity index, expressed as $-\sum_{i=1}^k p(c_i) \log_2 p(c_i)$, where $p(c_i)$ denotes the proportion of pixels of color c_i and k is the total number of unique colors in the histogram. High color diversity indicates a rich variety of colors, which typically contributes to the visual complexity and aesthetic quality of the image.

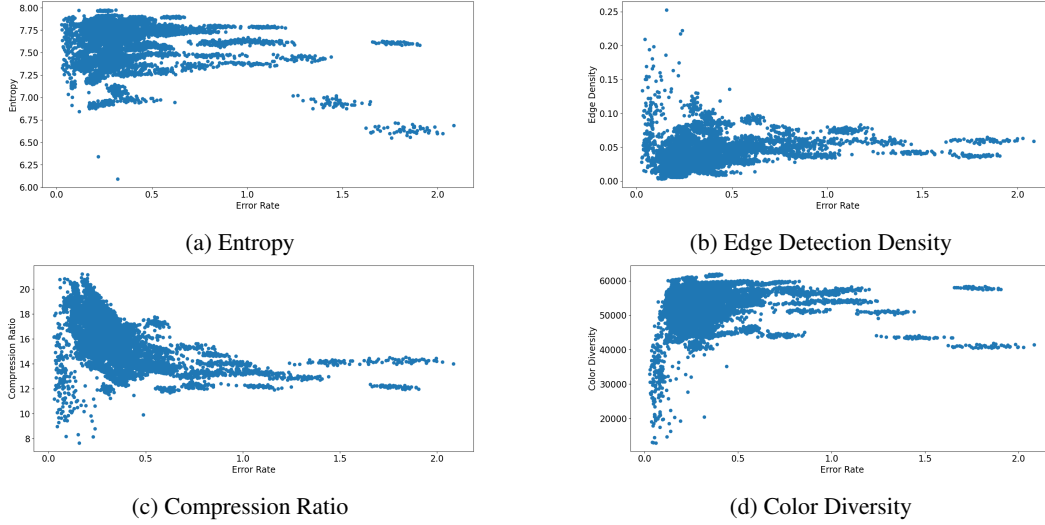


Figure 13: Image complexity metrics vs error rate

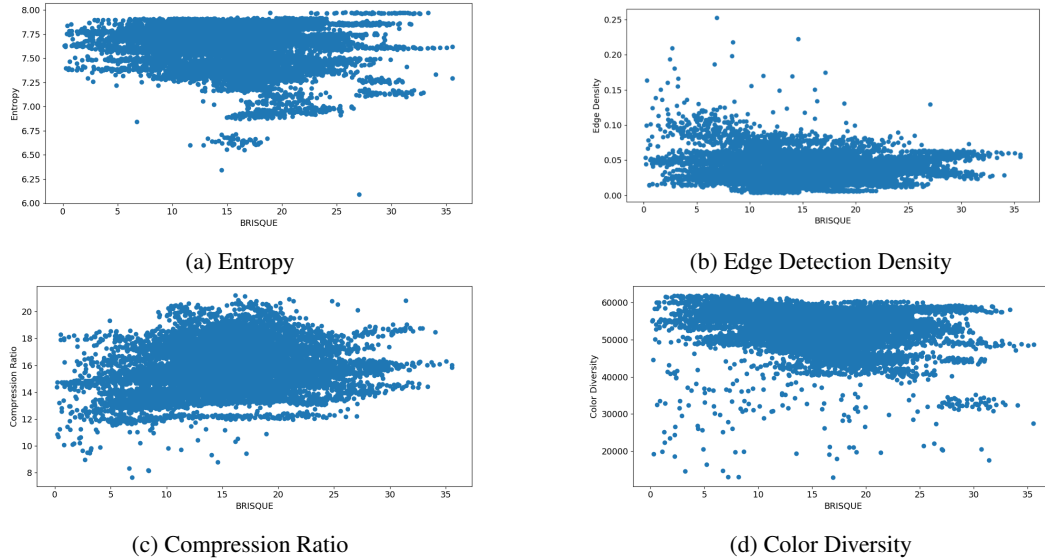


Figure 14: Image complexity metrics vs BRISQUE

While we do not observe a perfect correlation across all metrics, we note consistent trends with certain measures such as entropy and edge density. Specifically, as entropy and edge density increase, both the error rate and BRISQUE scores tend to decrease. We hypothesize that optimizing for error rate influences certain image characteristics, such as entropy and edge density, which are also associated with BRISQUE scores. This relationship may partially explain why we observe improved BRISQUE scores even though our primary focus is on minimizing error rate. For the other metrics, namely compression ratio and color diversity, the patterns are not as clear.

I Computational time

We show the average time required to optimize images using the DDIM-based cover selection method in Table 6 for both the CelebA-HQ and AFHQ-Dog datasets. We calculate computation time by determining the number of DDIM backward sampling steps required to achieve the lowest message recovery error, and then multiplying that number by the average duration of each step. As anticipated, for the CelebA-HQ dataset, the average computation time increases with the payload size. A similar

trend is observed in the AFHQ-Dog dataset; however, an exception occurs at a payload of $B = 4$ bpp. This anomaly can be attributed to the fact that, as shown in Table 2, the DDIM-optimized images for this payload do not exhibit a significantly lower error rate compared to the original images. All experiments were conducted using a NVIDIA A-100 GPU.

Table 6: Average computation time of DDIM-based cover selection (in seconds) for different payload values.

Dataset	1 bpp	2 bpp	3 bpp	4 bpp
CelebA-HQ	0.69	6.85	11.52	24.83
AFHQ-Dog	0.04	0.91	5.33	0.34

J Steganalysis: detailed settings and additional experiments

We adopt the simulation settings outlined in Chen et al. [2022] for our experiments. **In Scenario 1**, the steganography model M is trained without specific techniques to avoid detection by steganalysis. We assume the attacker, who performs steganalysis, knows the architecture of M but has no access to its weights, training data, or hyperparameters. However, the attacker can train a surrogate model M' to generate their own steganographic images. To simulate this scenario, we trained a steganalysis model on the CelebA dataset and used it to detect steganographic images generated from the AFHQ-Dog dataset. Interestingly, detection rates did not consistently decrease with lower payload sizes, a phenomenon also noticed in LISO Chen et al. [2022], on which our framework is based. We hypothesize this behavior arises from the distributional mismatch between training and testing data, as discussed earlier. **In scenario 2**, We leverage the fact that neural steganalysis methods are entirely differentiable, and that LISO uses gradient-based optimization. This allows us to reduce security risk by incorporating an additional loss term from the steganalysis system into the LISO optimization process. Specifically, during evaluation, if an image is identified as steganographic, we add the logit value of the steganographic class to the loss function.

In addition to XuNet (Xu et al. [2016]), we compute the steganalysis results of SRNet, another state-of-the-art steganalysis system (Boroumand et al. [2018]). The results of both schemes are compared in Table 7. Our observations indicate that the images generated by our framework effectively resist steganalysis by SRNet. This is evidenced by the significant drop in detection rate when transitioning from scenario 1 to scenario 2. As a reminder, in scenario 2, we exploit the differentiability of the steganalyzer (SRNet) and incorporate an additional loss term to account for steganalysis.

Table 7: Steganalysis results with SRNet and XuNet.

	Payload B	SRNet Det. (%) ↓		XuNet Det. (%) ↓	
		Original	DDIM	Original	DDIM
Scenario 1	1 bpp	15	13	37.1	37.5
	2 bpp	14.5	32.5	31.34	15.42
	3 bpp	61.5	55.5	20.39	34.82
	4 bpp	76	76	97.37	97.35
Scenario 2	1 bpp	0.0	0.0	0.0	0.0
	2 bpp	0.0	0.0	0.0	0.0
	3 bpp	0.0	0.0	3.2	2.1
	4 bpp	2	1	9.2	8.6

K Robustness to Gaussian noise

In this section, we evaluate the robustness of our DDIM-based approach to Gaussian noise. The experimental setup remains the same as described in Section 3.1 and illustrated in Fig. 2, with the only modification being the injection of Gaussian noise, distributed as $\mathcal{N}(0, \beta)$, into the output of the steganographic encoder. The decoder subsequently processes the noisy steganographic image to estimate the embedded message. Results of this experiment are presented in Table 8. Our findings demonstrate that the proposed framework produces cover images resilient to Gaussian noise,

Table 8: Robustness to Gaussian noise for a payload $B = 4$ bpp on CelebA-HQ.

Variance β	Error Rate (%) \downarrow		BRISQUE \downarrow		SSIM \uparrow		PSNR \uparrow	
	Original	DDIM	Original	DDIM	Original	DDIM	Original	DDIM
0.01	2.2	1.9	12.1	12.8	0.62	0.63	26.85	27.6
0.02	7.1	6.5	15.98	14.33	0.57	0.56	25.75	26.34
0.03	12.3	11.8	15.01	14.38	0.56	0.58	25.73	26.32

achieving lower error rates while preserving high visual quality. This is confirmed by visual quality metrics such as BRISQUE, SSIM, and PSNR, which remain comparable to those of the original images. An intriguing future direction is to extend this setup to handle perturbations beyond Gaussian noise. One approach could involve pretraining the LISO steganographic encoder-decoder pair under such conditions before applying our framework. Alternatively, our method could be applied to steganographic frameworks other than LISO, particularly those explicitly designed to handle image perturbations, such as Tancik et al. [2020] and Bui et al. [2023].

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: The claims made in the introduction and abstract, such as advantages in message recovery and image quality, as well as the analogy to the waterfilling algorithm are all described in Sections 3.3, 4, 5.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: We discuss two limitations of our work: 1) we noticed some image quality decline at lower bpp levels in CelebA-HQ and AFHQ-Dog, and discussed some regularization methods that we tried (Section 5.1 and Appendix C). 2) In the conclusion, we mention that we focused on independent Gaussian channels in Section 4, and that we would like to extend the setup to correlated channels in future work.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory Assumptions and Proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: No theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental Result Reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Yes, we described detailed training procedures in Section 3 and Appendix B.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We mention all the datasets and pretrained models that we used. Our code will be made public upon acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental Setting/Details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Yes, we described all the training details and hyperparameters in Section 3 and Appendix B.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment Statistical Significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Error bars were not reported for computational reasons and time constraints.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments Compute Resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We describe the time required for our method as well as the resources in Appendix I.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code Of Ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: Our research follows the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader Impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: We discuss the main uses of Steganography in Section 1, and the benefits of our cover selection approach under practical settings in Section 5.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our work doesn't pose such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Yes, we cite all the existing assets throughout the paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: Yes, we describe our algorithm in Section 3 and Appendix B. Our code will be released upon acceptance.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: Our work didn't involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: our work does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.