Coverless Multi-Image Steganography Utilizing GANs & Diffusion Models

D.D Hettiarachchi

Coverless Multi-Image Steganography Utilizing GANs & Diffusion Models

D.D. Hettiarachchi

Index No : **20000782**

Supervisor: **Dr. P.V.K.G Gunawardana**

June 2025

Submitted in partial fulfillment of the requirements of the
B.Sc in Computer Science Final Year Project (SCS4224)

Declaration

I certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a degree or diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organizations.

Candidate Name : D.D. Hettiarachchi

[illegible]

Signature of Candidate **Date: 2025.06.30**

This is to certify that this dissertation is based on the work of Mr.Dineth Hettiarachchi under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Supervisor Name : Dr. P.V.K.G. Gunawardana

[illegible]

Signature of Supervisor **Date:** 30-June-2025

Acknowledgement

I would like to thank all those who supported me in carrying out this research. This includes my supervisor Dr. Kasun Gunawardana and my friends who kept me motivated during the process.

Abstract

This thesis proposes a novel framework for coverless multi-image steganography that leverages the complementary strengths of Generative Adversarial Networks and diffusion models. Traditional steganographic systems often depend on explicit cover modification, suffer from low capacity, or lack robustness against detection. Diffusion models excel in image synthesis but remain underexplored for multi-image steganography

The proposed system successfully reconstructs two RGB secret images from generated container images with minimal perceptual loss, achieving PSNR values of up to 17 dB and SSIM scores of exceeding 0.6 even under standard Gaussian noise and JPEG compression. The hiding capacity reached 48 bits per pixel while maintaining resistance to steganalysis. This demonstrates the practical viability of the hybrid approach for secure, scalable, and robust coverless steganography.

Limitations of prompt sensitivity, domain dependency, and recovery failure under high image similarity are identified, and future work explores flow-based conditioning to enhance recovery consistency. The findings demonstrate the viability of combining GAN and diffusion paradigms in scalable, multi-image, coverless steganographic systems.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

1.1.1 Steganography

Steganography is the practice of concealing messages within other non-secret text or data, ensuring that the presence of the message remains undetectable to unintended recipients. Unlike cryptography, which obscures the content of a message, steganography hides the existence of the message itself. Historically, methods of steganography have included techniques such as writing with invisible ink.

In the digital realm, steganography often involves embedding information within digital media files, such as images, audio, or video. Techniques include modifying the least significant bits (LSBs) of pixel values in images or altering audio samples in a way that is imperceptible to human senses. The goal is to ensure that the presence of hidden information is undetectable to unintended recipients.

Advanced steganographic techniques can be categorized into three primary strategies: (J. Liu et al. 2020)

- **Cover Modification**: Involves embedding data within a cover medium (e.g., an image) in such a way that minimizes distortion and preserves the statistical properties of the original cover. This approach aims to conceal the modifications effectively, making detection by steganalysis tools more challenging.

- **Cover Selection**: Entails establishing a mapping between specific features of the cover medium and the secret message. By selecting cover mediums based on these mappings, this method aims to enhance security and reduce the likelihood of detection.

- **Cover Synthesis**: Involves generating new cover mediums specifically designed to hide information. Techniques such as Generative Adversarial Networks (GANs) have been employed to synthesize cover images that effectively conceal secret data, thereby improving the security and robustness of steganographic systems.

Coverless Steganography refers to methods that do not modify a carrier medium to embed information. Instead, these techniques exploit inherent features of the medium, such as pixel brightness, color, texture, or high-level semantics, to directly represent secret data. The advantage of this approach is that conventional steganalysis tools cannot detect the presence of hidden information, as there is no altered cover medium to analyze. Cover synthesis and cover selection methods are considered coverless, as they generate or select cover mediums specifically designed to conceal data without modification.

1.1.2 Steganalysis

Steganalysis is the process of detecting hidden information within digital media. It serves as the counterpart to steganography, aiming to identify, extract, or destroy concealed messages. Unlike cryptanalysis, which deals with deciphering encrypted messages, steganalysis focuses on detecting the very existence of hidden information.

Techniques in steganalysis can be broadly categorized into signature-based and statistical methods. Signature-based steganalysis searches for known patterns or anomalies introduced by specific steganographic tools. Statistical steganalysis, on the other hand, involves analyzing the statistical properties of media files to detect deviations from expected norms, which may indicate the presence of hidden data.

Advanced steganalysis methods employ machine learning algorithms, including deep learning, to improve detection accuracy. These approaches train models on large datasets of clean and steganographically altered media to identify subtle differences that may not be apparent through traditional analysis.

1.2 Motivation

Traditional steganographic techniques, especially those relying on explicit cover modification, are becoming increasingly susceptible to detection by modern steganalysis tools. Furthermore, existing deep learning-based methods often focus on hiding a single image and frequently require a host image as a carrier. These constraints limit both the security and scalability of current systems.

At the same time, the advent of powerful generative models—particularly GANs and diffusion models—has revolutionized image synthesis, enabling the generation of high-quality, photorealistic content without reliance on a predefined cover. Despite their success in image generation tasks, their potential for secure, scalable steganography remains underexplored.

This research is motivated by the gap between these two domains: the stagnation of multi-image steganography and the underutilization of generative models in constructing robust, coverless steganographic systems. By combining GANs for structural fidelity with diffusion models for semantic guidance and key-conditioned synthesis, this work aims to design a next-generation steganographic pipeline capable of hiding multiple images without relying on any explicit carrier, while remaining resistant to standard detection methods.

1.3 Research Gap

While diffusion models have demonstrated state-of-the-art performance in image synthesis tasks due to their iterative denoising process and high fidelity outputs, their application in digital image steganography, especially when fused with GAN-based architectures remains underdeveloped. Recent work such as S. Xu et al. (2024) has shown that diffusion models can perform image-to-image translation while maintaining cycle consistency. However, this capability has not been exploited in the context of multi-image steganographic encoding.

Existing steganographic approaches, including the Cam-GAN framework proposed by Lakshan (2024), typically rely on CycleGANs to embed secret images into stego outputs. These GAN-only methods often suffer from limited visual realism and susceptibility to artifacts due to training instability and mode collapse. Enhancements via ESRGAN or Pix2Pix can improve output fidelity, but remain isolated techniques rather than components of an integrated system.

Moreover, there is currently **no unified generative framework** that combines the strengths of diffusion models and GANs for **multi-image coverless steganography**. Existing methods either rely on single-image payloads, explicit cover modification, or do not scale to multi-image scenarios without degradation. Prior diffusion-based techniques such as Y. Xu, X. Zhang, et al. (2024) still depend on embedding strategies rather than full synthesis, thus failing to meet the definition of coverless steganography.

This research addresses the above limitations by developing a hybrid architecture that leverages both GANs and diffusion models to generate visually realistic container images capable of embedding and recovering multiple secret images, without using an explicit cover. This novel combination aims to improve imperceptibility, increase steganographic capacity, and resist detection.

1.4 Research Questions

1. How can a hybrid generative architecture combining GANs and diffusion models be used to construct a multi-image coverless steganographic pipeline?

2. How does the use of diffusion-enhanced image synthesis affect the visual quality and imperceptibility of stego images compared to GAN-only meth-

ods?

1.5 Aims and Objectives

1.5.1 Aim

To design, implement, and evaluate a hybrid generative framework that combines GANs and diffusion models for multi-image coverless steganography, with the goal of improving visual realism, steganographic capacity, and resistance to detection.

1.5.2 Objectives

- To design a steganographic pipeline that integrates GAN-based modules with diffusion models for image synthesis and refinement.
- To evaluate the visual quality of stego images using standard image quality metrics such as PSNR, SSIM.
- To assess the accuracy of secret image reconstruction under varying experimental conditions and noise levels.
- To compare the hybrid model's performance against GAN-only baselines in terms of steganographic imperceptibility and resistance to steganalysis.
- To analyze architectural and optimization challenges in integrating GAN and diffusion models into a unified pipeline.

1.6 Scope

- The study focuses on digital image steganography, specifically the development of **multi-image coverless** methods.
- The system synthesizes stego images without requiring an explicit cover image.

- The research evaluates both image quality and steganographic robustness using quantitative metrics.
- Comparative evaluation will be performed against other steganographic schemes.

1.7 Research Methodology

This research adopts a design–implement–evaluate methodology, grounded in empirical experimentation. The study was structured into the following key phases:

1.7.1 Model Design and Hypothesis Formulation

Several architectural designs were conceptualized to investigate multi-image coverless steganography. The central hypothesis posited that a hybrid framework integrating diffusion models and GANs could outperform conventional GAN-based approaches in terms of imperceptibility, hiding capacity, and reconstruction fidelity. Four primary architectural variants were developed and explored:

- A Convolutional Variational Autoencoder (VAE)
- A ResNet-based GAN
- A modified ESRGAN
- A Pix2PixGAN (final model)

1.7.2 Dataset Preparation

Custom datasets were generated by blending image pairs utilizing the `addWeighted()` function in OpenCV. Additional datasets tailored for ESRGAN training incorporated Gaussian noise and blur to simulate diffusion-induced distortions. A subset of each dataset was excluded from the training process and reserved for evaluation purposes.

1.7.3 Model Training and Implementation

Model training was conducted using the ANT PC, Google Colab and Kaggle platforms. For the diffusion process we utilize the CompVis/stable-diffusion-v1-4 pre-trained diffusion model (Rombach et al. 2022) and to guide the diffusion model in image generation and noising tasks we use choose to use prompts (descriptions of secret and cover images) as the public and private keys.

The training strategies implemented were:

- Standard adversarial training for GAN architectures, employing PatchGAN discriminators
- Reconstruction and Kullback–Leibler (KL) divergence loss for the VAE
- A combined loss comprising L1 loss, adversarial loss, and perceptual loss for ESRGAN and Pix2Pix

Hyperparameters were empirically optimized based on loss and qualitative output assessment. However, fine tuning was limited due to resource constraints (timing and GPU usage) on Kaggle and Google Colab.

1.7.4 Evaluation Procedure

Model evaluation encompassed both qualitative and quantitative metrics, including:

- **Steganographic Imperceptibility:** Natural Image Quality Evaluator (NIQE) and visual evaluation
- **Hiding Capacity:** Measured in bits per pixel (BPP)
- **Reconstruction Accuracy:** Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), Mean Squared Error (MSE) and Semantic Accuracy (via visual inspection)

- **Robustness:** Evaluated under Gaussian noise and JPEG compression using Learned Perceptual Image Patch Similarity (LPIPS) and other degradation metrics

The proposed models were then benchmarked against existing methods.

Chapter 2

Literature Review

2.1 Foundations of Image Steganography

Classical methodologies generally fall under three paradigms: *cover modification*, *cover selection* and *cover synthesis* (J. Liu et al. 2020). Techniques based on cover modification typically use pixel-level adjustments such as LSB substitution or distortion minimization (Holub, Fridrich, and Denemark 2014), but they often introduce statistical anomalies detectable by steganalysis tools. Cover selection strategies tend to be low-capacity, while early cover synthesis approaches were limited by the lack of photorealistic image generators.

2.2 Generative Adversarial Networks and Their Role

The emergence of GANs (Goodfellow et al. 2014) introduced a new class of data-driven models capable of learning to generate images indistinguishable from real data. This adversarial training framework consisting of a generator and a discriminator enabled significant progress in producing realistic images, which was quickly adopted by the steganography community to address limitations in traditional systems.

GANs have since been used in three primary ways within steganography:

- **GAN-based Cover Modification**: GANs assist in generating more secure cover images or learning adaptive embedding strategies (Volkhonskiy, Borisenko, and Burnaev 2016).
- **GAN-based Cover Selection**: This strategy uses GANs to map information to pre-existing natural images, avoiding direct image modification (Ke et al. 2019).
- **GAN-based Cover Synthesis**: Information is embedded directly in synthetic images generated from noise or latent variables (Z. Zhang et al. 2019; D. Hu et al. 2018).

2.2.1 Advancements and Applications

A variety of architectures have emerged. SGAN and SSGAN employ adversarial learning to generate cover images less susceptible to detection (Volkhonskiy, Borisenko, and Burnaev 2016; Shi et al. 2018). Other frameworks, such as ASDL-GAN and UT-SCA-GAN, aim to learn embedding cost functions, making message insertion more secure (Yang, Ruan, et al. 2019; Yang, K. Liu, et al. 2018).

End-to-end systems like SteganoGAN integrate encoder-decoder architectures with critic networks to jointly optimize message recovery and visual fidelity (K. A. Zhang et al. 2019). Unsupervised and semi-supervised variants such as SWE (D. Hu et al. 2018) and ACGAN-based schemes (M.-m. Liu et al. 2017) leverage label control or noise-to-image mapping to synthesize stego images without relying on specific covers.

2.2.2 Limitations of GANs

While GANs have demonstrated significant potential in synthesizing visually plausible stego-images, a few limitations hinder their effectiveness in steganographic applications.

One of the primary challenges is mode collapse, wherein the generator converges to producing a limited subset of outputs regardless of the input noise vector. This reduction in generative diversity can diminish the variability among stego-images, thereby increasing the risk of detection by steganalysis algorithms (*Mode collapse — Wikipedia, The Free Encyclopedia* 2024).

Furthermore, GAN-generated container images do not consistently achieve high perceptual realism. Especially in the current day where the usage of diffusion models are becoming prevalent.

Such limitations undermine the imperceptibility of GAN-based steganographic schemes, as visually anomalous or structurally inconsistent images are more susceptible to detection by both automated and manual inspection.

2.3 Use of Diffusion Models

Historically, image generation for steganographic applications has predominantly relied on GANs. However, recent advancements in diffusion models have led to their adoption in the domain of steganography due to their improved generative quality and controllability.

Jois, Beck, and Kaptchuk (2023) introduced a novel approach wherein the variance noise inherent in the diffusion process was utilized as a medium for embedding hidden information. This method demonstrated the feasibility of using the noise space of diffusion models as an effective steganographic channel.

Yu et al. (2024) further advanced this paradigm by incorporating Low-Rank Adaptation (LoRA) and ControlNet mechanisms in conjunction with diffusion models. Their approach enabled the controlled generation of stego-images through Denoising Diffusion Implicit Models (DDIM) inversion, facilitating invertible image translation.

Inspired by this, Y. Xu, X. Zhang, et al. (2024) proposed a hierarchical, multi-image steganographic framework based on diffusion models. Similar to Yu et al. (2024), their method employed public and private keys to guide the embedding and extraction processes, thereby enhancing the security and robustness of the steganographic system.

2.4 Related Work

Early approaches to deep learning-based steganography primarily focused on embedding a single image within another of equal dimensions. Baluja (2017) pioneered this direction by proposing a convolutional neural network capable of hiding a full-color image within another image of the same size, establishing the feasibility of end-to-end learned steganographic frameworks. However, this approach was quite vulnerable to steganalysis.

Subsequent research extended this concept toward more challenging configurations. L. Liu et al. (2022) explored the inverse scenario by attempting to embed a larger color image within a smaller host image, pushing the boundaries of information density in steganographic embedding.

In parallel, the concept of coverless steganography gained traction. X. Liu et al. (2020) introduced Cam-GAN, a GAN-based framework capable of generating full-sized stego-images without the need for a pre-existing cover image. This approach was further extended by Dharmawimala (2023), who demonstrated the embedding of two grayscale images into a single stego-image using a refined version of Cam-GAN. Finally Lakshan (2024) built upon this work by enhancing the model's capacity to embed two colored images. While more embedding capacity was achieved, the model did not prove to be robust, and was incapable of generating realistic cover images.

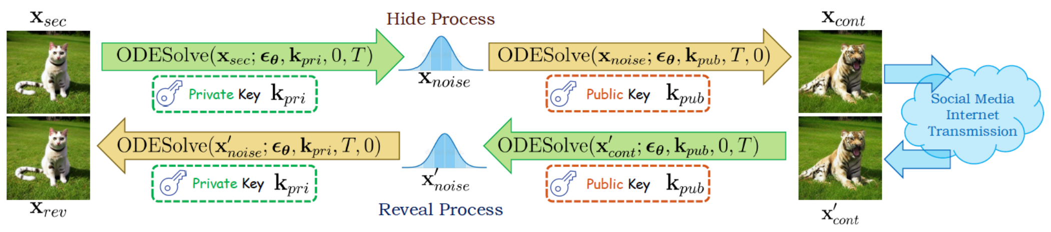
$\mathbf{x}_{sec}$
Hide Process
$\mathbf{x}_{cont}$
ODESolve($\mathbf{x}_{sec}; \boldsymbol{\epsilon_\theta}, \mathbf{k}_{pri}, 0, T$)
Private Key $\mathbf{k}_{pri}$
$\mathbf{x}_{noise}$
ODESolve($\mathbf{x}_{noise}; \boldsymbol{\epsilon_\theta}, \mathbf{k}_{pub}, T, 0$)
Public Key $\mathbf{k}_{pub}$
Social Media Internet Transmission
ODESolve($\mathbf{x}'_{noise}; \boldsymbol{\epsilon_\theta}, \mathbf{k}_{pri}, T, 0$)
Private Key $\mathbf{k}_{pri}$
$\mathbf{x}'_{noise}$
ODESolve($\mathbf{x}'_{cont}; \boldsymbol{\epsilon_\theta}, \mathbf{k}_{pub}, 0, T$)
Public Key $\mathbf{k}_{pub}$
$\mathbf{x}_{rev}$
Reveal Process
$\mathbf{x}'_{cont}$

Figure 2.1: Coverless image steganography framework used in Yu et al. (2024)

The introduction of diffusion models into image steganography enabled new capabilities in security, robustness, and controllability. One notable example is the CRoSS framework proposed by Yu et al. (2024), which uses pretrained conditional diffusion models to implement a fully coverless steganographic pipeline without requiring any additional training or modification to the diffusion backbone. At the core of CRoSS is the use of deterministic Denoising Diffusion Implicit Models (DDIM), which enable an invertible mapping between image space and latent noise through an ODE-based formulation. This property allows for encoding a secret image into latent space using DDIM inversion and reconstructing it with high fidelity, assuming the correct conditions (i.e., textual prompts) are provided.

CRoSS exploits this mechanism by introducing a dual-key system, where prompts serve as public and private keys. During the hide process, a secret image is first transformed into a latent representation using a private key prompt, then decoded into a visually plausible container image using a public key prompt. The reveal process reverses this operation: the container is re-encoded with the public key and decoded back to the secret using the private key. Importantly, this architecture ensures that the container image does not explicitly embed or alter any low-level information from the secret image, making it resistant to statistical or perceptual steganalysis.

In addition to the invertibility advantage, CRoSS leverages community tools such as ControlNets and LoRAs to enhance the controllability of the container image generation process, supporting diverse use cases without retraining. However, CRoSS is designed for single-image hiding and relies solely on pre-existing diffusion checkpoints. It does not address the problem of multi-image embedding.

Y. Xu, X. Zhang, et al. (2024) build upon the approach proposed by Yu et al. (2024) by extending the CRoSS diffusion scheme to support the hierarchical embedding of two secret images. In their method, the first secret image is transformed into a container image using the original CRoSS diffusion process. Subsequently, the second secret image is embedded within the resulting container image, thereby achieving a two-tiered steganographic structure.

Compared to earlier methods that focused on either enhancing embedding capacity (L. Liu et al. 2022; Lakshan 2024) or eliminating the need for cover images (X. Liu et al. 2020; Yu et al. 2024), recent approaches like CRoSS and its extensions

(Y. Xu, X. Zhang, et al. 2024) emphasize robustness, invertibility, and semantic controllability. However, these diffusion-based models are often constrained to single-image hiding or require sequential embeddings with limited scalability. Our method bridges this gap by combining the generative strength of GANs with the precision of diffusion-based reconstruction, enabling efficient and robust multi-image embedding without relying on sequential processing or pre-trained diffusion checkpoints alone.

Chapter 3

Design and Implementation

3.1 Preliminary Studies with Diffusion Steganography

3.1.1 Single-Image Steganography

The objective of this preliminary study was to assess the effectiveness of a pre-trained diffusion model for single image steganography in the absence of additional postprocessing layers. Specifically, the investigation focused on evaluating the model's robustness across diverse image characteristics, including variations in subject distinctiveness, stylistic similarity, semantic domains and the fidelity of the reconstructed secret image. For this we follow the approach used by Yu et al. (2024) in their steganographic scheme

When applied to single-image steganography tasks, the diffusion model was able to generate plausible container images that retained sufficient information to allow for the extraction of the secret image. However, performance varied across different images and conditions. Notably, the model exhibited improved effectiveness when the secret images contained distinct and well-defined subjects as can be seen in Figure 3.1. This suggests that the diffusion model benefited from salient structural features, which were easier to embed and retrieve during the generation and reconstruction processes.

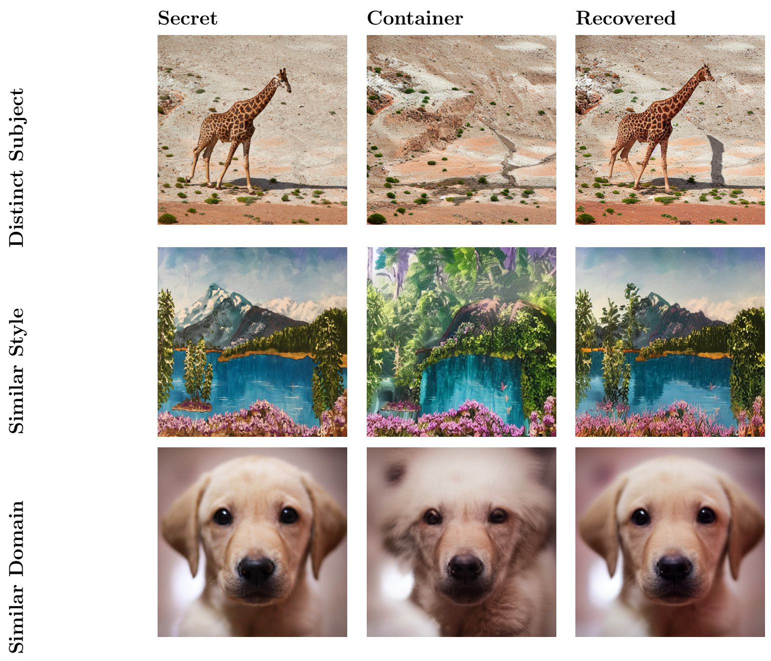

Figure 3.1: Comparison of secret, container, and recovered images under different conditions

Additionally, the diffusion model showed stronger performance when there was a high degree of stylistic or domain similarity between the secret image and the target container image. For instance, hiding an oil painting-style secret image within a container image of a similar artistic style yielded superior results, both in terms of visual quality and reconstruction fidelity. A particularly illustrative case was when the model was tasked with converting a secret image of a puppy into a container image of a wolf, both belonging to the same semantic domain.

3.1.2 Multi-Image Steganography

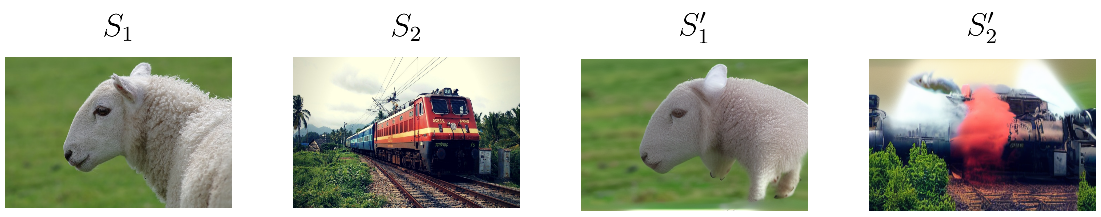

Figure 3.2: Results obtained by concatenating secret images

We then extended our experiments to multi-image steganography, where we tested a concatenation approach wherein two images were joined side by side and treated as a single composite input for the diffusion-based steganographic model. However, this method proved ineffective due to the lack of independent guidance for the diffusion model during the reconstruction process of each image.

We utilized a combined textual description of both images as the private key, but this led to suboptimal results in secret image recovery. The features of the individual secret images were not fully reconstructed, and often the image with a more salient or easily describable subject dominated the reconstruction. Furthermore, when a more complex key was used in an attempt to describe both images in detail, the resulting output often exhibited blending artifacts between the two images. This blending severely hindered our ability to accurately extract the embedded secret information from either image. An illustration of these results is shown in Figure 3.2 where S_1, S_2 refer to the two secret images being hid, and S_1', S_2' refer to the recovered secret images.

3.2 Approach 1 - Utilizing a Variational Autoencoder

3.2.1 Rationale

The design of the proposed steganographic framework was guided by the need to balance image concealment, latent representation fidelity, and reconstructability. Several considerations informed each stage of the pipeline:

Latent Fusion via Convolutional VAE. The decision to use a convolutional VAE stems from the need to jointly encode two image representations into a compact latent space. The VAE provides a structured probabilistic framework that naturally supports latent fusion and disentanglement. Additionally, convolutional layers were introduced to exploit the spatial structure of image data, improving feature extraction and leading to better generalization and reconstruction performance. The ability of VAEs to learn smooth and continuous latent representations was critical for maintaining useful encodings that could be passed through subsequent stages.

Our model architecture was partially inspired by the work of Baluja (2017), who proposed an encoder-decoder network to perform image-in-image steganography. In Baluja's method, a host image and a secret image were passed through a deep convolutional encoder-decoder pipeline that allowed the hidden content to be reconstructed with high fidelity. Motivated by this approach, we adopted a similar architecture to support the joint encoding and later separation of multiple secret images.

Channel-Wise Concatenation and Shared Encoding. Concatenating the transformed inputs along the channel dimension allowed the encoder to treat both images as part of a single input structure, enabling the network to learn correlated features. This design encourages the VAE to preserve information relevant to both images while enabling individual recovery in the decoding stage. The shared latent space representation **im** serves as a flexible medium for both fusion and downstream transformation.

3.2.2 Proposed Steganographic Method

This method involves a two-stage transformation and combination scheme designed to obscure the original images while enabling their seamless extraction. The process is as follows:

In the first stage, each secret image, denoted as S_1 and S_2, undergoes a partial transformation (noising) using a pre-trained diffusion model (DM). The key modulates the noising process, effectively acting as a reversible transformation key.

$$S_{p1} = \mathcal{D}_{K_{\text{priv}}^{(1)}} S_1, \quad S_{p2} = \mathcal{D}_{K_{\text{priv}}^{(2)}} S_2.$$

This transformation obscures the original visual features of the image while preserving an encoded representation. The transformed images and S_1 and S_2 are combined using a dual-input convolutional VAE. The encoder concatenates the inputs along the channel axis and maps them into a shared latent representation

im. This design ensures that both image features are jointly encoded while still allowing separable decoding.

$$\mathbf{im} = \text{Encoder}(S_{p1}, S_{p2})$$

In the second stage, the fused latent representation **im** is further processed using a separate public key K_{pub}. This processed latent representation is then passed through the diffusion model $\mathcal{D}$ to generate the final container image $\mathcal{C}$:

$$\mathcal{C} = \mathcal{D}_{K_{\text{pub}}}(\mathbf{im}).$$

For retrieval, the container image $\mathcal{C}$ undergoes inverse processing using the public key K_{pub} to reconstruct the latent representation $\mathbf{im}'$ of the two original images:

$$\mathbf{im}' = \mathcal{D}^{-1}_{K_{\text{pub}}}(\mathcal{C}).$$

The latent representation is then decoded using the Decoder to obtain the two latent representations of the individual secret images.

$$S'_{p1}, S'_{p2} = \text{Decoder}(im')$$

The reconstructed latent tensors are then individually processed using the same diffusion model $\mathcal{D}$, each guided by its respective private key $K^{(1)}_{\text{priv}}$ and $K^{(2)}_{\text{priv}}$. This step enables the full restoration of the original secret images S'_1 and S'_2:

$$S'_1 = \mathcal{D}^{-1}_{K^{(1)}_{\text{priv}}}(S'_{p1}), \quad S'_2 = \mathcal{D}^{-1}_{K^{(2)}_{\text{priv}}}(S'_{p2}).$$

In this approach, we attempted to leverage a Convolutional Variational Autoencoder to combine two latent image representations into one tensor and then decode them individually. The VAE was expected to enable the transformation of images into a compact latent space representation, allowing for efficient fusion and separation of multiple images.

3.2.3 Model Architecture

The overall architecture consists of an encoder and a decoder network, both based on convolutional layers. The encoder maps the input image to a latent space,

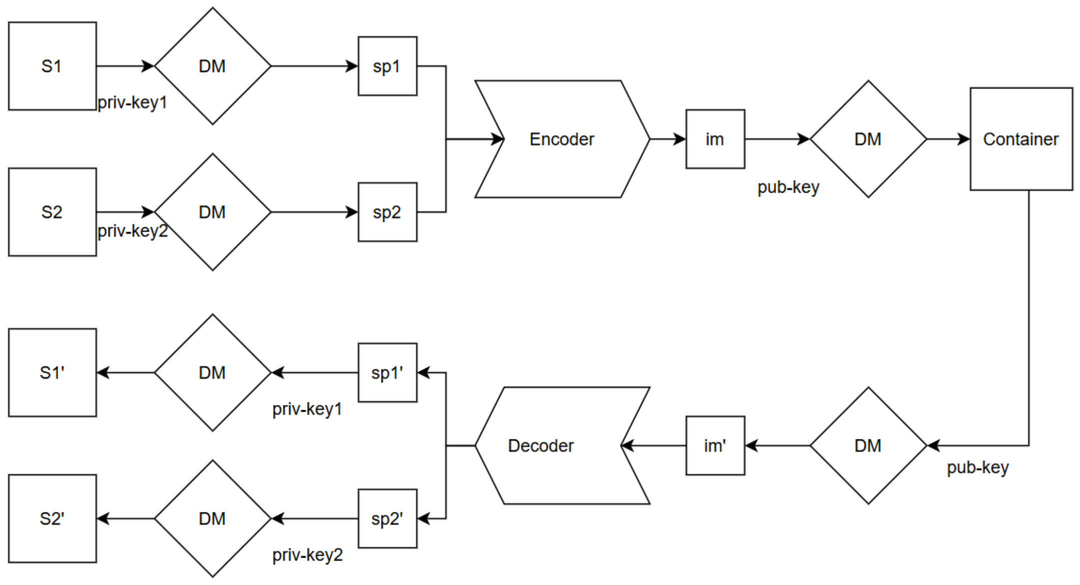

Figure 3.3: Steganographic scheme with VAE

while the decoder reconstructs the original image from the latent representation.

Encoder Architecture

The encoder processes the input image by passing it through a series of convolutional layers, progressively reducing spatial dimensions while increasing the number of feature channels. The encoder architecture is as follows:

- The input image, with three color channels, is passed through three convolutional layers. These layers use increasing filter sizes (16, 32, and 64 channels, respectively). The kernel size is set to 4, the stride to 2, and padding to 1 for each layer.

- After each convolutional layer, a ReLU activation function is applied, introducing non-linearity to the learned features.

- Following the convolutional layers, the feature map is flattened and passed through two fully connected layers to produce two outputs: the mean vector and the log-variance vector. This step ensures that the latent space follows a Gaussian distribution, which is critical for the reparameterization trick used in VAEs.

Decoder Architecture

The decoder reconstructs the image from the latent space representation using transposed convolutional layers. The decoding process follows these steps:

- The latent vector is passed through a fully connected layer to reshape it into the appropriate spatial dimensions.
- Three transposed convolutional layers are then used to progressively upsample the feature map back to the original image dimensions (128 $\times$ 64).
- The final layer applies a sigmoid activation function to ensure the pixel values of the reconstructed image lie within the range $[0, 1]$.

3.2.4 Data Preparation

For this experiment, we used the Animals10 dataset, which consists of approximately 28,000 images. These images were categorized into several classes, including dog, horse, elephant, butterfly, chicken, cat, cow, sheep, and squirrel. The labels associated with each class were used as the private key for processing each image. A dataset of processed images, transformed using these private keys, was created for training and experimentation.

3.2.5 Experiment 1

Training Procedure

The model is optimized using a loss function that consists of two components:

- **Reconstruction Loss ($\mathbf{L}_{\text{reconstruction}}$):** This term measures the difference between the original and reconstructed image, computed using Mean Squared Error. The L2 loss for an image at the pixel level is given by:

$$\mathcal{L}_{\text{pixel}} = \frac{1}{N}\sum_{i=1}^{N}(\hat{I}_i - I_i)^2 \tag{3.1}$$

where: $\hat{I}_i$ is the predicted pixel value at position i, I_i is the original pixel value (ground truth) at position i, N is the total number of pixels in the image.

- Kullback-Leibler (**KL**) Divergence: This regularization term ensures the learned latent distribution is close to a standard normal distribution, $N(0, I)$, preventing overfitting and encouraging a smooth latent space.

The total loss is defined as:

$$L = L_{\text{reconstruction}} + \beta \cdot \text{KL}$$

where β is a weighting factor (set to 1 in this experiment). This combined loss function allows the model to learn both to reconstruct the input images accurately and to maintain a well-behaved latent space representation.

In this experiment, the VAE is trained for 100 epochs with a batch size of 32. We use the ANT PC for this training, utilizing the GeForce RTX 2080 Ti GPU with 11GB VRAM, and the process took approximately two days. In each iteration, two images are randomly selected from the dataset, one from each class (e.g., cat and dog). These images are passed through the encoder to obtain their latent representations, which are then decoded back into image space using the decoder. The gradients are computed with respect to the loss, and the model parameters are updated using the Adam optimizer.

The key training parameters are summarized in Table 3.1.

Parameter	**Value**
Optimizer	Adam
Learning Rate	1×10^{-4}
KL Divergence Weighting (β)	1.0

Table 3.1: Experiment 1 parameters

3.2.6 Results and Analysis

During the first training process, we observed that the loss function did not decrease beyond a certain point, and the VAE was unable to adequately reconstruct the latent representations of the input images at various stages. This suggested that the model was not learning sufficiently from the data.

In response, we enhanced the architecture by introducing convolutional layers to the VAE (resulting in the model described above), expecting it to better capture spatial hierarchies and improve reconstruction. Upon retraining the model, we saw an improvement in performance as the loss continued to decrease more effectively.

However, despite the improvements, we encountered another issue when we attempted to reconstruct the images using a diffusion model. Specifically, when the latent representations decoded through the VAE were passed into the diffusion model, it failed to accurately reconstruct the two secret images. Figure 3.4 shows two secret images and their recovered counterparts respectively. This indicated that while the convolutional VAE was able to learn better representations, the integration with the diffusion model did not yield the expected results.

We hypothesize that this failure is due to the diffusion noising process destructively altering the essential features encoded in the VAE latent space. This likely limited the number of features the VAE could encode and thereby preventing faithful recovery.

While VAE provided a clean latent fusion method, its Gaussian prior bottlenecked fine detail recovery. We discarded it in favor of ResNet-based GANs due to their stronger spatial feature retention,

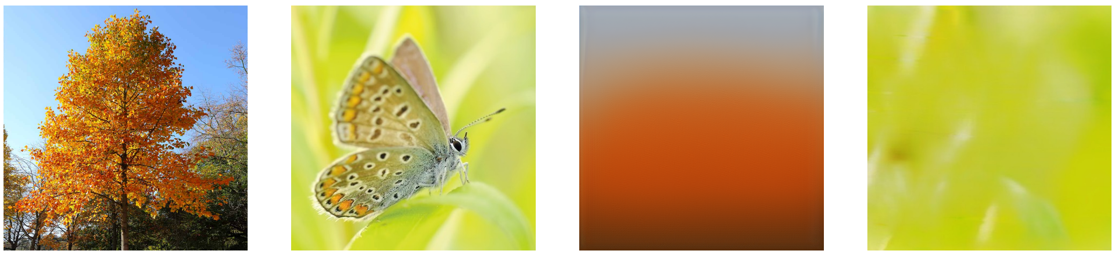
Figure 3.4: Images processed via the trained VAE

3.3 Approach 2 - Utilizing a ResNet Generator

3.3.1 Rationale

The design of this steganographic method was informed by several motivations and previous advancements in the field. The primary challenge addressed is the extraction of two secret images from a single fused image, while maintaining both high image quality and effective separation of the individual components. To meet this challenge, the architecture and training strategy were developed with the following considerations:

Use of a Generator with ResNet Blocks: The generator architecture incorporates residual blocks, chosen for their established ability to preserve high-level features and facilitate deep network training. Residual connections help alleviate the vanishing gradient problem and support the retention of spatial detail—critical for tasks requiring image decomposition or separation.

This architectural choice builds on the work of X. Liu et al. (2020), who applied a similar structure in a steganographic context to extract a single hidden image. Subsequently, Lakshan (2024) extended this concept to recover two full-resolution RGB images from a single composite image. Drawing inspiration from these approaches, we adopted and modified a comparable generator design to extract two secret images. We hypothesised that the residual learning mechanism would support better disentanglement of overlapping visual features, thus enhancing recovery performance. This rationale guided the network architecture and the loss function design within this experimental framework.

Use of the PatchGAN discriminator: The choice of the PatchGAN architecture for the discriminator is motivated by its efficacy in enforcing high-frequency local realism in generated images. Unlike traditional discriminators that evaluate entire images, PatchGAN assesses overlapping $N \times N$ patches, focusing on local features and textures. This localized evaluation compels the generator to produce outputs with authentic local details, resulting in sharper and more convincing images (Isola et al. 2017). Furthermore, Lakshan (2024) employed the same PatchGAN architecture in his steganographic scheme with success.

3.3.2 Proposed Steganographic Method

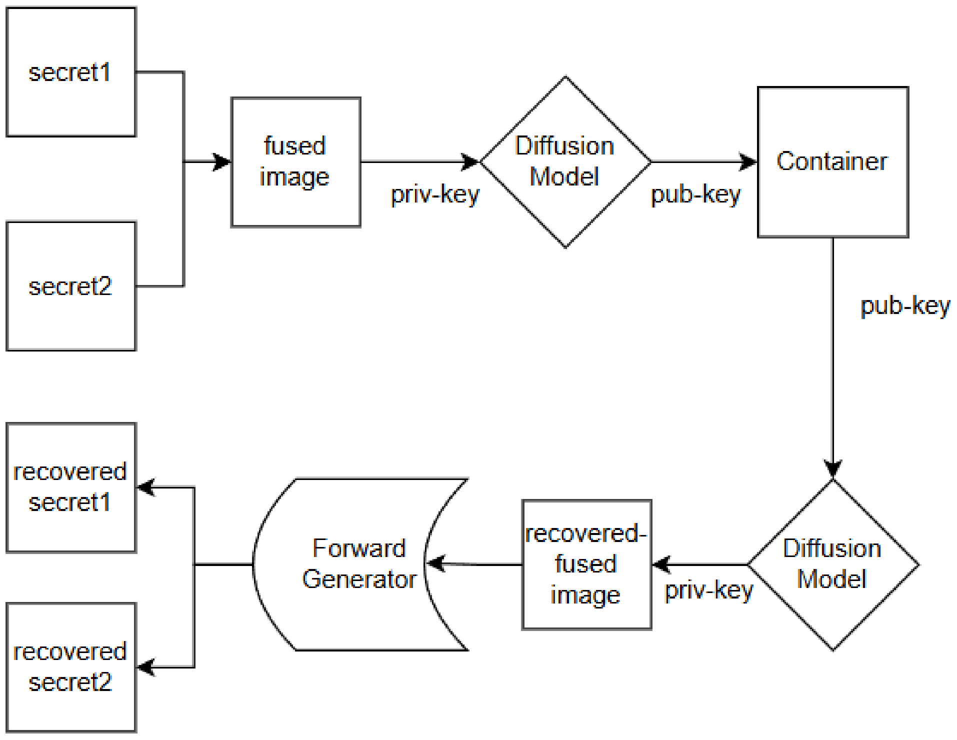

Figure 3.5: Steganographic scheme with ResNet Generator

The scheme begins with the fusion (linear blend) of two secret images, denoted as S_1 and S_2 into a single image $\mathbf{F}(S_1, S_2) = F_s$, which forms the initial entity for further processing.

Next, the fused image F_s is processed using a diffusion model $\mathcal{D}$, guided by the private key K_{priv}. This step obfuscates the underlying features of the fused image, producing an intermediate image. The output of this diffusion process is further processed using the public key K_{pub}, resulting in the generation of the container image $\mathcal{C}$:

$$\mathcal{C} = \mathcal{D}_{K_{\text{pub}}, K_{\text{priv}}}(F_s).$$

In the retrieval phase, the container image $\mathcal{C}$ is first processed using the public key K_{pub}, and the resulting image is then subjected to further processing using the private key K_{priv}. This step retrieves the fused image F'_s:

$$F'_s = \mathcal{D}^{-1}_{K_{\text{priv}}}(\mathcal{D}^{-1}_{K_{\text{pub}}}(\mathcal{C})).$$

Finally, the fused image F'_s is separated using the Generator $\mathcal{G}$, which extracts and outputs the two original secret images S'_1 and S'_2:

$$S'_1, S'_2 = \mathcal{G}(F'_s).$$

3.3.3 Model Architecture

ResNet Generator Architecture

The model architecture is designed to take an input image and generate two separate images. This is achieved through the use of convolutional, residual, and transposed convolutional layers. Below, we describe the architecture in detail:

The model is structured in the following stages:

1. Preprocessing and Downsampling: The input image, which has dimensions (C_{in}, H, W) (where C_{in} is the number of input channels, and H and W are the height and width of the image), first undergoes padding with a 3-pixel reflection padding. Then, it is passed through a 7x7 convolution layer with filters, followed by batch normalization and a ReLU activation. The image is downsampled using two layers of 3x3 convolutions with stride 2 and padding 1, each followed by batch normalization and a ReLU activation. This results in a reduction in spatial resolution.

2. Residual Blocks: The model utilizes residual blocks, which help to preserve features across multiple layers. These blocks consist of two 3x3 convolutions with batch normalization and ReLU activations, with a skip connection between the input and output.

3. Further Downsampling: After the residual blocks, the image undergoes additional downsampling through a few layers, each consisting of a 3x3 convolution followed by batch normalization and ReLU activation. This further reduces the spatial resolution of the feature map.

4. Bottleneck and Feature Fusion: The resulting feature map is passed through another network, which consists of more residual blocks and convolutions. This bottleneck stage captures high-level features and fuses them into a latent representation, which is used to generate two distinct output images.

5. Upsampling: The feature map is then passed through an upsampling process using transposed convolution layers. These layers gradually increase the spatial resolution of the feature map, returning it to the original input resolution. Each transposed convolution is followed by batch normalization and ReLU activation.

6. Postprocessing and Output Generation: Finally, the upsampled feature maps are passed through two different postprocessing streams, which produce the final two output images. Both streams consist of reflection padding, a 7x7 convolution, and a Tanh activation function to ensure the outputs are in the range $[-1, 1]$.

This architecture allows the model to extract and separate two distinct images from a single input image, capturing different aspects of the input through the use of residual learning and feature fusion.

Discriminator Network

The Discriminator network is a convolutional neural network designed to distinguish between real and generated (fake) images, providing feedback to the generator to help improve the realism of generated images. It follows a PatchGAN architecture, classifying overlapping image patches as real or fake. The architecture consists of four convolutional blocks, each incorporating batch normalization and Leaky ReLU activation. The network starts with 64 channels and gradually increases to 512 channels in the final convolutional layer, which produces a single output channel using a Sigmoid activation function to yield the probability of a patch being real.

Key Components of the Discriminator

- **Convolutional Layers:** The discriminator consists of several convolutional layers with LeakyReLU activations. These layers progressively reduce the spatial dimensions of the input image while increasing the number of feature maps.
- **Final Output:** The final output of the discriminator is a single scalar value representing the probability that the input image is real (close to 1) or fake (close to 0).

Loss functions

Discriminator Loss The discriminator's loss consists of two parts:

- **Real Loss ($\mathbf{L_{real}}$):** This term encourages the discriminator to classify real images as real. It is calculated as:

  $$\mathcal{L}_{\text{real}} = \log(D(I_{\text{real}}))$$

  where $D(I_{\text{real}})$ is the discriminator's output for a real image.

- **Fake Loss ($\mathbf{L_{fake}}$):** This term encourages the discriminator to classify generated images as fake. It is calculated as:

  $$\mathcal{L}_{\text{fake}} = \log(1 - D(\hat{I}_{\text{fake}}))$$

  where $D(\hat{I}_{\text{fake}})$ is the discriminator's output for a generated image.

The total discriminator loss is the average of the real and fake losses:

$$\mathcal{L}_D = \frac{1}{2}\left(\mathcal{L}_{\text{real}} + \mathcal{L}_{\text{fake}}\right) \tag{3.2}$$

Generator Loss The Generator's loss function is a weighted sum of two components:

- **Reconstruction Loss** ($\mathbf{L}_{\text{reconstruction}}$): This term measures the difference between the original and reconstructed image, computed using Mean Absolute Error. The L1 loss for an image at the pixel level is given by:

$$\mathcal{L}_{\text{pixel}} = \frac{1}{N}\sum_{i=1}^{N} |\hat{I}_i - I_i| \tag{3.3}$$

  where: $\hat{I}_i$ is the predicted pixel value at position i, I_i is the original pixel value (ground truth) at position i, N is the total number of pixels in the image.

- Adversarial Loss ($\mathbf{L}_{\text{adversarial}}$) : as calculated in 3.2

The reconstruction loss is weighted by a factor α, and the Discriminator loss is weighted by a factor β. Therefore, the total Generator loss is given by:

$$L_{\text{Generator}} = \alpha L_{\text{reconstruction}} + \beta L_{\text{D}} \tag{3.4}$$

3.3.4 Data preparation

The vehicles and landscapes datasets were employed. The `addWeighted()` function from the OpenCV library was utilized to create a dataset of linearly blended images, with an alpha value of 0.5. This process resulted in a paired dataset comprising three distinct components: a vehicle image, a landscape image, and the corresponding blended image. In total, approximately 5,000 blended images were generated for subsequent analysis.

3.3.5 Experiment 2

Training Process

The model training process employed a GAN architecture, with the ResNet Generator and the Discriminator. The Generator takes a blended image as its input and produces two distinct output images. These generated images are then evaluated by the same Discriminator, which classifies each image as either real or fake, based on its realism. Both the Generator and the Discriminator were trained concurrently, at the same rate.

The Discriminator, in turn, is trained to minimize the binary cross-entropy loss, which measures its ability to correctly classify real and fake images. This adversarial training procedure encourages the Generator to produce more realistic images, while the Discriminator becomes more adept at distinguishing between the two.

The model was trained over 100 epochs, with both the Generator and the Discriminator being updated at each epoch. The training was conducted on Kaggle with an NVIDIA Tesla P100 GPU with 16GB VRAM and the training process took approximately 12 hours.

Parameter	Value
α	10
β	100

Table 3.2: Experiment 2 parameters

3.3.6 Experiment 3

Training Process

In this experiment, we employed a similar GAN architecture as in Experiment 2 but modified the training schedule for the Generator and Discriminator by training the Generator twice as often as the Discriminator. Specifically, the Generator weights were updated every epoch, while the Discriminator weights were only updated every odd epoch. The goal of this modification was to evaluate the effect of more frequent updates to the Generator on the quality of the generated images. The remaining parameters were left unchanged from Experiment 2. The training was conducted on Kaggle with an NVIDIA Tesla P100 GPU with 16GB VRAM. The final training process took approximately 11 hours.

3.3.7 Experiment 4

Training Process

In this experiment, we modified the architecture further by introducing two separate Discriminators (with the same architecture as before), one for each of the two output images generated by the Generator. Each Discriminator was tasked with evaluating the realism of its corresponding generated image. This setup aimed to explore the effect of having specialized Discriminators for each output, with the hypothesis that it could improve the quality of the generated images.

The Generator was trained to minimize the weighted sum of two loss components similar to Experiment 2. The reconstruction loss (equation 3.3) and the sum of the Adversarial losses from both Discriminators where each Discriminator's loss function was calculated as per equation 3.2. The total Generator loss was therefore modified as follows:

$$L_{\text{Generator}} = \alpha L_{\text{reconstruction}} + \beta_1 L_{\text{adversarial}_1} + \beta_2 L_{\text{adversarial}_2}$$

where $L_{\text{adversarial}_1}$ and $L_{\text{adversarial}_2}$ are the binary cross-entropy losses computed by the first and second Discriminators, respectively, and β_1 and β_2 are the weights for each Discriminator's contribution to the total loss.

The Discriminators were trained similarly to Experiment 2, with each aiming to minimize its own binary cross-entropy loss.

The model was trained for 100 epochs, with both the Generator and the Discriminators being updated according to their respective schedules. The training was conducted on Kaggle with an NVIDIA Tesla P100 GPU with 16GB VRAM and the training process took approximately 13 hours.

Parameter	Value
α	10
β_1	50
β_2	50

Table 3.3: Experiment 4 parameters

3.3.8 Results and Analysis

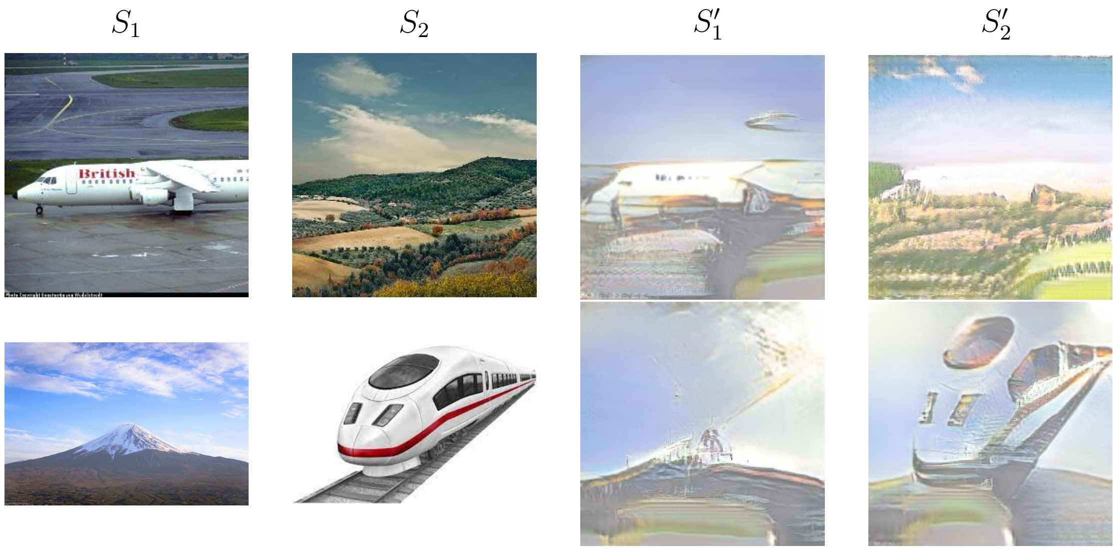

Figure 3.6: Images recovered using the ResNet generator

During the training process of Experiment 2, the Generator's loss did not decrease significantly beyond a certain point, and the intermediate images generated at regular intervals during the training process lacked realism. To address this, we adjusted the weight of the reconstruction loss (α) by reducing it in comparison to the adversarial loss (β), as initially, the reconstruction loss had a greater weight. After applying the parameters outlined in Table 3.3.5, we observed a slight improvement in the realism of the generated images, although the reconstruction accuracy remained suboptimal.

We hypothesized that the insufficient weight assigned to the reconstruction loss, coupled with the continuous decrease in the Discriminator's loss, contributed to this issue. In Experiment 2, the Generator's loss plateaued and then remained erratic within a given range, while the Discriminator's loss continued to decrease. This suggested that the Discriminator was becoming too proficient, learning too rapidly, which might have hindered the Generator's progress. To mitigate this, we modified the training schedule in Experiment 3 by updating the Generator twice as often as the Discriminator. Even with the adjusted schedule, the Generator's loss remained stagnant, suggesting limited learning progression. This could have been due to the Generator not receiving sufficient feedback from the Discriminator.

Additionally, when evaluating the Generator, we noted that the two generated images appeared too similar, and the Generator failed to properly separate the fused image into two distinct outputs.

In an attempt to address these issues in Experiment 4, we introduced two separate Discriminators, each tasked with evaluating the realism of one of the two generated images. This modification was made with the assumption that having specialized Discriminators would improve performance by providing more specific feedback for each image domain. The Generator was trained to minimize a loss function that included both reconstruction loss and the individual losses from each Discriminator. Although the issue of the Generator's erratic loss persisted, the results in Experiment 4 showed some improvement (determined by visual inspection) over Experiment 2 and Experiment 3 in terms of separating the fused image into two distinct outputs.

The results of extracting the two secret images can be seen in Figure 3.6, where the model trained in Experiment 4 was used. S_1 and S_2 represent the secret images, and S_1' and S_2' represent the recovered secrets.

Given that this approach was unable to reconstruct the secret images convincingly, we decided to explore a different steganographic approach for recovery.

3.4 Approach 3 - Utilizing a ESRGAN

3.4.1 Rationale

The rationale for utilizing an Enhanced Super-Resolution Generative Adversarial Network (ESRGAN) in the proposed steganographic method is grounded in its demonstrated ability to enhance perceptual image quality and remove visual artifacts, both of which are critical for recovering intelligible secret images after diffusion-based transformations.

Handling Residual Artifacts: The fusion and diffusion steps in our steganographic scheme introduce spatially distributed artifacts that degrade the quality of the recovered images. Traditional models often struggle with perceptual quality, especially in recovering fine texture and details. ESRGAN, as proposed by X. Wang et al. (2018), addresses this issue by integrating a Residual-in-Residual Dense Block generator architecture, which has shown superior performance in refining degraded images while maintaining structural coherence and sharpness. Its residual learning capacity makes it suitable for filtering out noise introduced by fusion and denoising steps in diffusion.

Perceptual Fidelity: Unlike pixel-wise loss-based methods that often result in blurry outputs, ESRGAN incorporates a perceptual loss that operates in a feature space derived from pretrained CNNs. In our setup, MobileNetV3 was selected over more conventional backbones like VGG or ResNet due to resource constraints. Despite its lightweight design, MobileNetV3 retains the ability to extract semantically rich features from images, enabling perceptual comparisons that guide the generator to prioritize human-recognizable fidelity. This design choice was motivated by findings from Howard et al. (2019) which highlight its efficiency in mobile and constrained compute environments.

3.4.2 Proposed Steganographic Method

The scheme begins with the fusion of two secret images, denoted as S_1 and S_2 into a single fused entity, $\mathbf{F}(S_1, S_2) = F_s$. Next, the fused image F_s is processed using a diffusion model $\mathcal{D}$, which is guided by two distinct keys: the public key K_{pub} and the private key K_{priv}. The private key should be a combination of the individual private keys of each secret image. This guidance transforms the fused image into the container image $\mathcal{C}$, which is defined as:

$$\mathcal{C} = \mathcal{D}_{K_{\text{pub}}, K_{\text{priv}}}(F_s)$$

In the retrieval phase, the container image $\mathcal{C}$ is passed through the diffusion model $\mathcal{D}$, guided by the public key K_{pub}. This step produces a partially recovered fused image $\mathcal{R}$:

$$\mathcal{R} = \mathcal{D}^{-1}_{K_{\text{pub}}}(\mathcal{C})$$

Then, the partially recovered image $\mathcal{R}$ is processed again using two separate private keys, $K_{\text{priv}}^{(1)}$ and $K_{\text{priv}}^{(2)}$, corresponding to the two secret images S_1 and S_2, respectively. This final diffusion step yields the partially recovered secret images:

$$S'_{p1} = \mathcal{D}^{-1}_{K_{\text{priv}}^{(1)}}(\mathcal{R}), \quad S'_{p2} = \mathcal{D}^{-1}_{K_{\text{priv}}^{(2)}}(\mathcal{R})$$

Afterwards, the partially recovered images must undergo a further refinement process using an ESRGAN. The ESRGAN step is performed as follows:

$$S'_1 = \text{ESRGAN}(S'_{p1}), \quad S'_2 = \text{ESRGAN}(S'_{p2})$$

After this step, the ESRGAN outputs S'_1 and S'_2, which are the extracted secret images that were initially fused together. This step is necessary to preserve the quality of the secret images after the diffusion-based transformations and to remove residual artifact.

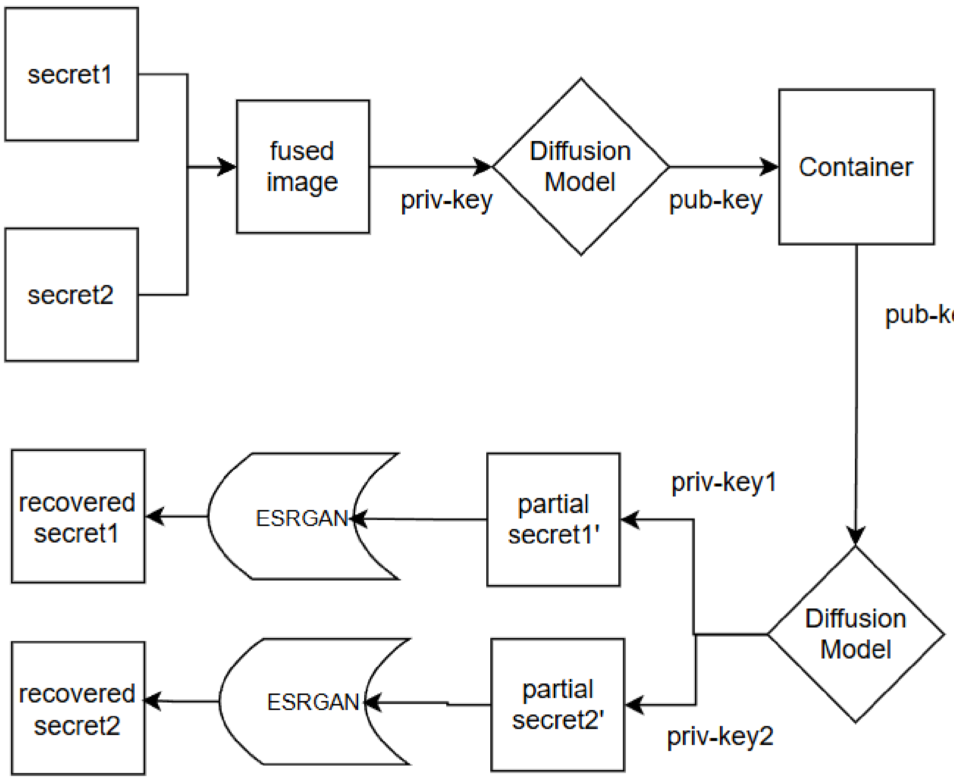

Figure 3.7: Steganographic scheme with ESRGAN

3.4.3 Model Architecture

ESRGAN Generator Architecture

The architecture presented in this work based off of a deep learning-based model for image super-resolution using GANs. The model aims to remove distortions and restore images by employing a Generator network, a Discriminator network, and perceptual and adversarial losses. We used the same Discriminator architecture as the previous experiments.

The Generator is the core component of the architecture, responsible for generating high-resolution images from low-resolution inputs. The generator utilizes a Residual in Residual Dense Block Network (RRDBNet). RRDBNet is designed for image super-resolution tasks, specifically for enhancing image details while removing artifacts during upscaling.

Key Components of the Generator

Residual Dense Blocks (RDBs): The generator consists of multiple Residual Dense Blocks stacked together. Each RDB operates in a hierarchical fashion, where each layer receives input from the previous layer, and each intermediate output is densely connected to the next. The output of each RDB is used as input for subsequent RDBs, allowing the network to progressively capture more complex image features. The RDB structure is as follows:

- Each RDB contains Convolutional Layers with a kernel size of 3×3 followed by LeakyReLU activations.
- Each intermediate output is concatenated with the next layer's input, which allows the network to capture dense feature representations.
- The final output of the RDB is passed through a convolutional layer that refines the features.

Up-sampling Layers: After the feature maps are processed through the stacked RDBs, the spatial resolution of the output is increased using two convolutional layers, followed by interpolation layers. The interpolation layer helps upscale the feature map by a factor of either 2x or 4x, depending on the configuration.

Final Output: The final output of the generator is a high-resolution image produced by a convolutional layer that ensures the output has the same number of channels as the ground truth (typically 3 channels for RGB images).

Feature Extractor (MobileNetV3)

A Feature Extractor based on MobileNetV3 is used to compute the perceptual loss. MobileNetV3 is a lightweight deep learning model pretrained on large-scale image datasets. It is used to extract high-level features that help compare the perceptual quality of generated and target images.

The features extracted by MobileNetV3 are used to compute the perceptual loss, ensuring that the generated images not only match the ground truth pixel-wise but also in terms of higher-level feature representations.

Loss Functions

Generator Loss The total loss for the generator is a weighted combination of the following components:

- **Reconstruction Loss :** L1 Loss (equation 3.3)
- **Perceptual Loss:** This loss is computed by comparing high-level feature representations extracted from a pretrained MobileNetV3 model. The perceptual loss ensures that the generated images have similar high-level features to the ground truth. The loss is defined as:

  $$\mathcal{L}_{\text{perceptual}} = \frac{1}{M} \sum_{i=1}^{M} ||\Phi(\hat{I}_i) - \Phi(I_i)||^2$$

  where:

  - M: The total number of images in the batch.
  - $\hat{I}_i$: The i-th generated image from the model.

  - I_i: The i-th ground truth image.
  - Φ: The feature extraction process (using the pretrained MobileNetV3) that maps the images to their high-level feature representations.

- Adversarial Loss ($\mathbf{L}_{\text{adversarial}}$) : equation 3.2

The total generator loss is the weighted sum of these losses:

$$\mathcal{L}_G = \lambda_{\text{pixel}}\mathcal{L}_{\text{pixel}} + \lambda_{\text{perceptual}}\mathcal{L}_{\text{perceptual}} + \lambda_{\text{adversarial}}\mathcal{L}_{\text{adversarial}}$$

where λ_{pixel}, $\lambda_{\text{perceptual}}$, and $\lambda_{\text{adversarial}}$ are the hyperparameters controlling the relative importance of each loss term.

Parameter	**Value**
λ_{pixel}	1
$\lambda_{\text{perceptual}}$	1
λ_{D}	1×10^{-3}

Table 3.4: Experiment 5 parameters

3.4.4 Data Preparation

The following process was carried out to create a dataset for training the ESRGAN to remove residual artifacts from the extracted secret images.

An image is randomly selected from either the CelebHQ dataset, which consists of high-resolution facial images, or the Vehicles dataset, which contains vehicle images. The selection determines which image is the dominating image ($\mathbf{Image}_{strong}$) and weaker image ($\mathbf{Image}_{weak}$). One image from the chosen dataset is then paired with a corresponding image from the other dataset, ensuring that each pair consists of one facial image and one vehicle image. Gaussian noise is added to $\mathbf{Image}_{strong}$ to simulate the noise typically encountered in a diffusion process. Additionally, slight blurring is also applied.

Finally, the transformed $\mathbf{Image}_{strong}$ is blended with $\mathbf{Image}_{weak}$. A linear blending operation is used where $\mathbf{Image}_{strong}$ is weighted by a scalar parameter α

and $\mathbf{Image}_{weak}$is weighted by $1-\alpha$, such that the blended image $\mathbf{I}_{blend}$ is expressed as:

$$\mathbf{I}_{blend} = \alpha\mathbf{Image}_{strong} + (1-\alpha)\mathbf{Image}_{weak}$$

The parameter α was set to 0.8 to ensure that the weaker image does not completely overshadow the stronger image features in the final blended image.

The resulting dataset simulates the image corruption process that occurs when the image is processed by the diffusion model in the proposed steganographic method.

3.4.5 Experiment 5

Training Process

Training was conducted on Kaggle using a NVIDIA Tesla P100 GPU with 16GB VRAM. The final training process, after experimentation, took approximately 13 hours.

During training, we experimented with different pre-trained models for feature extraction in the perceptual loss calculation. Initially, we tested VGG-19 and ResNet-50 due to their strong feature extraction capabilities. However, the high memory consumption of these models led to out-of-memory errors, which severely restricted the batch size that could be used. This resulted in excessively long training times per epoch, making it impractical to train the model given the limited resources.

After multiple trials, we settled on MobileNetV3, a lightweight feature extractor that provided a good balance between efficiency and performance. MobileNetV3 significantly reduced training time per epoch while maintaining reasonable perceptual quality in the reconstructed images.

The training was performed by feeding a corrupted image from the CelebHQ dataset and the corresponding original image into the Generator. The objective was to train the Generator to reconstruct the original image from the corrupted input. This process was repeated over 100 epochs, allowing the model to progressively learn to remove artifacts introduced by the diffusion process. For this the Adam optimizer was used with a learning rate schedule controlled by Cosine Annealing.

3.4.6 Results and Analysis

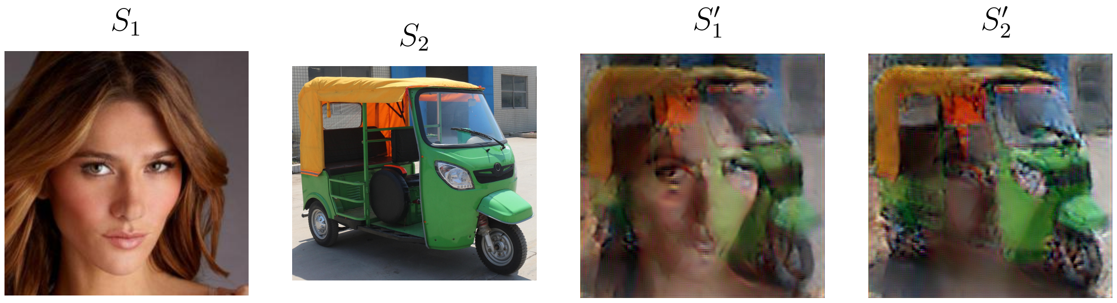

Figure 3.8: Images recovered using the ESRGAN

Despite the theoretical capability of the ESRGAN to remove distortions and create high resolution images, the method did not fully succeed. While the recovered secret images retained better color fidelity than previous ResNet-based outputs (Figure 3.6), they still contained leftover artifacts as seen in Figure 3.8. The ESRGAN was designed to address this issue by refining the recovered images, but it fell short in handling the kinds of distortions introduced by the fusion and diffusion recovery process.

We believe the residual artifacts in the recovered images can were too complex while the ESRGAN's design is primarily aimed at pixel-level enhancement and perceptual features, which may not have been suitable for handling complex distortions.

However, one positive development from this experiment is that the recovery pathway in the steganographic method used to extract two secrets from the container image is a novel approach. This method demonstrated a better preservation of extraction quality than other approaches, offering a promising direction for future improvements in steganographic recovery processes. The method, although not perfect, exhibited robust extraction quality in comparison to the previous experiments, which could be vital for improving overall recovery accuracy in complex steganographic systems

Although ESRGAN improved perceptual quality, it likely failed to correct semantic drift in high-entropy domains such as CelebHQ. This suggests that GAN-based refinement alone is insufficient for content-aware recovery

3.5 Approach 4 - Utilizing a Pix2Pix GAN

3.5.1 Rationale

The decision to adopt a Pix2Pix GAN for this steganographic method is based on a combination of practical needs and architectural strengths of the framework. Pix2Pix is a conditional GAN architecture introduced by Isola et al. (2017) and is widely known for its performance in image-to-image translation tasks. This is directly aligned with the goal of mapping an intermediate stego image back to its original form, making it a natural fit for our problem.

Firstly, the Pix2Pix architecture employs a U-Net generator with skip connections that enable the preservation of low-level spatial features. This is critical in our use case, where maintaining fine details and structural consistency in the image is essential to accurately recover hidden information without introducing detectable distortions.

Secondly, the loss function design used in Pix2Pix, combining L1 reconstruction loss with adversarial loss, provides a strong inductive bias toward both realism and fidelity. This balance helps to ensure that reconstructed images do not just look plausible, but also remain close to the original secret image in terms of pixel-wise accuracy.

Min-ha-zul Abedin and Yousuf (2023) introduced *StegoPix2Pix*, a steganographic technique built on the Pix2Pix architecture. Their results demonstrated that the method can achieve high accuracy in secret message recovery, while also providing resilience against detection by steganalysis tools. Their approach inspired the architecture shown in Figure 3.9, which integrates these strengths into the framework for our steganographic task.

Overall, Pix2Pix offers a compelling trade-off between architectural complexity, training efficiency, and empirical effectiveness. Its proven success in structured image translation and its demonstrated utility in related steganographic applications form the basis for its selection in this work.

3.5.2 Proposed Steganographic Method

We reuse the steganographic scheme from section 3.4.2 which previously showed strong performance. The only difference being the final step is performed as follows

$$S'_1 = \mathrm{Pix2Pix}_1(S'_{p1}), \quad S'_2 = \mathrm{Pix2Pix}_2(S'_{p2})$$

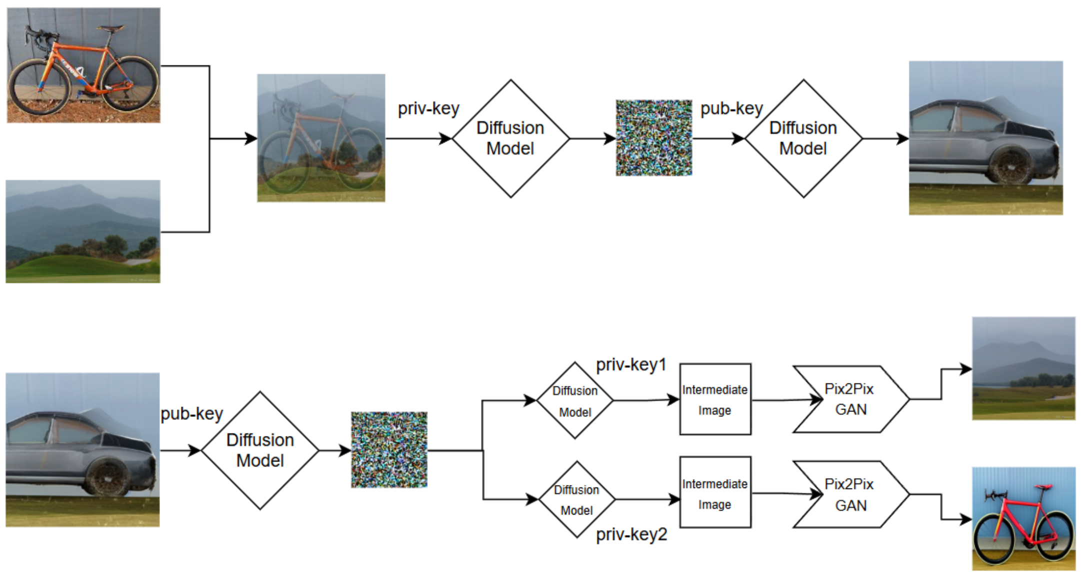

Figure 3.9: Steganographic scheme with Pix2Pix GAN

3.5.3 Model Architecture

The Pix2Pix model consists of two primary components: the generator and the discriminator, designed for image-to-image translation tasks. We used the same Discriminator architecture as the previous experiments.

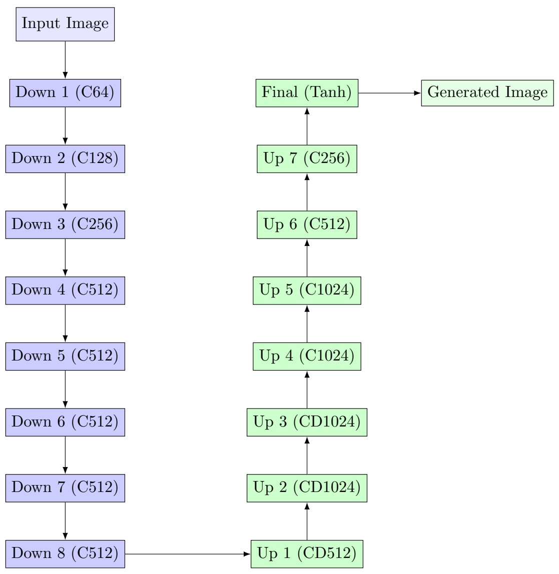

Figure 3.10: Pix2Pix Generator Architecture

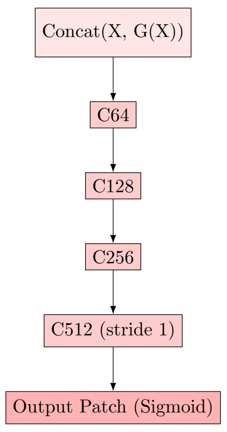

Figure 3.11: PatchGAN Discriminator Architecture

Generator

The generator employs a U-Net architecture, featuring an encoder-decoder structure with skip connections. The encoder is composed of eight downsampling layers, where each layer performs a convolutional operation followed by optional batch normalization and Leaky ReLU activation. The number of output channels progressively increases from 64 to 512 across the first seven layers, with the eighth layer maintaining 512 channels to capture the complex features of the input image.

Conversely, the decoder consists of seven upsampling layers that utilize transposed convolutions. Each upsampling layer concatenates features from the corresponding encoder layer through skip connections, preserving spatial information. The output channels decrease from 512 to 64, ultimately producing the output image in the final layer, which applies a transposed convolution followed by a Tanh activation function to achieve the desired output range.

Loss Functions

For training, the Pix2Pix model utilizes two loss functions. The generator loss function is a combination of the reconstruction loss (L1 loss) and the adversarial loss, similar to equation 3.4. The discriminator loss computes the average of the real and fake losses based on the discriminator's predictions, as per equation 3.2.

$$\mathcal{L}_G = \lambda_{L1}\mathcal{L}_1 + \lambda_{adversarial}\mathcal{L}_{\text{adversarial}}$$

3.5.4 Experiment 6

Training Procedure

The same datasets described in Sections 3.4.4 and 3.3.4 were used for training the models. Each training sample consists of an intermediate image and its corresponding original image, which are input into the Pix2Pix GAN. The objective of the model is to reconstruct the original image from the intermediate representation.

Training was conducted for 100 epochs on Kaggle using an NVIDIA Tesla P100 GPU with 16 GB of VRAM. The Adam optimizer was used, with a learning rate of 2×10^{-4}. Two Generators were trained following this approach.

Parameter	**Value**
λ_{L1}	100
$\lambda_{\text{adversarial}}$	1

Table 3.5: Experiment 6 parameters

3.5.5 Results and Analysis

Compared to previous approaches, the Pix2Pix GAN demonstrated significantly improved performance in recovering the secret images. The recovered images exhibited accurate color representation without any notable distortions. Furthermore, the majority of the image features were preserved, maintaining a high level of fidelity to the original input.

However, one observed limitation of the approach lies in the behavior of the Diffusion model, which introduced some stylistic alterations to the reconstructed images. The reconstructions were often overly smooth, losing natural texture in certain cases. Additionally, the model exhibited some difficulty in accurately recreating the background details, a challenge that was also highlighted in our preliminary studies. This issue seemed more pronounced when reconstructing complex or vast backgrounds, such as those found in landscape images, as opposed to more structured and recognizable objects like faces or vehicles.

The improved performance when reconstructing images featuring faces or vehicles can likely be attributed to the relatively uniform appearance of such objects. In contrast, landscapes are inherently more variable and expansive, making them more difficult for the model to capture and accurately reproduce. This discrepancy underscores the challenges the model faces when attempting to reconstruct larger and more complex visual contexts, as the model is more adept at handling smaller, more structured objects with consistent features.

Discussion and Justification We hypothesize that the superior performance of the Pix2Pix + Diffusion configuration arises from the synergy between Pix2Pix's supervised learning and the generative capabilities of the diffusion model. Unlike previous GAN variants (e.g., ResNet-based or ESRGAN), Pix2Pix is trained on paired data, allowing the model to learn a direct mapping from the fused input to the corresponding secret image outputs. This explicit supervision likely improved convergence and structural fidelity. Additionally, Pix2Pix's U-Net architecture—with its skip connections—facilitates the retention of low-level spatial details, which is critical in steganographic recovery. When combined with the semantic realism and noise tolerance provided by diffusion models, this architecture demonstrates both high reconstruction accuracy and robustness under perturbations such as Gaussian noise and JPEG compression. This could explain its superior PSNR and SSIM metrics observed during evaluation.

Overall, while the Pix2Pix GAN provided promising results in the context of this steganography scheme, there remains some room for improvement, particularly in handling complex background scenes and mitigating the smoothing effect introduced by the diffusion model.

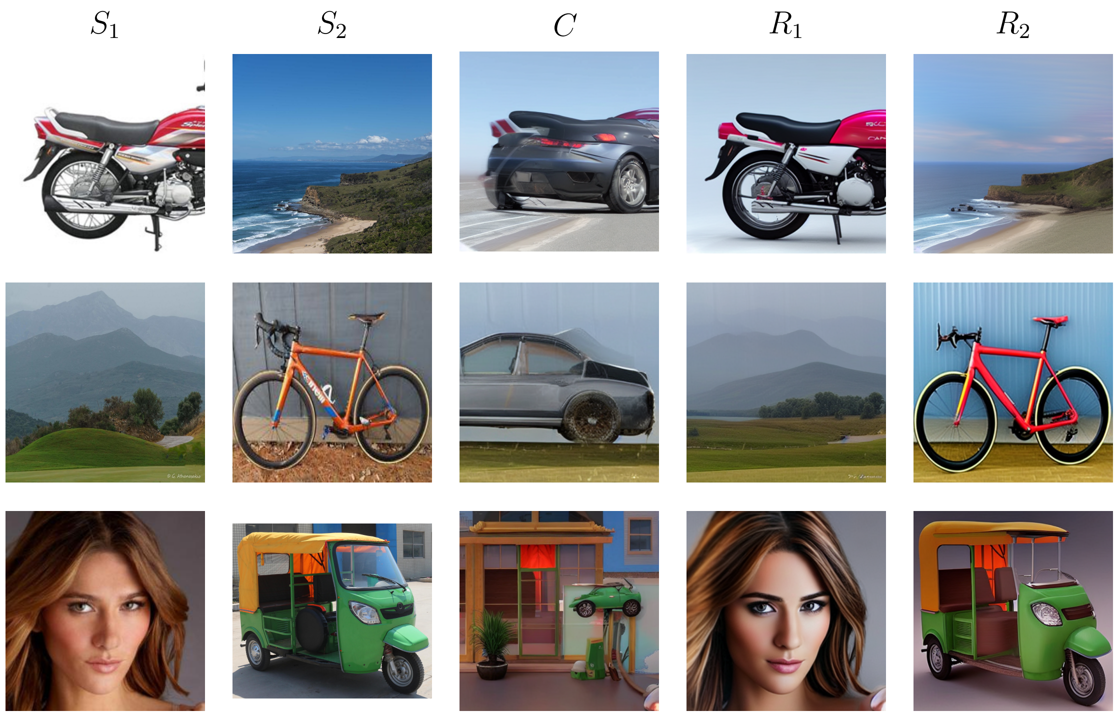

Figure 3.12: Images recovered using the Pix2Pix GAN

Chapter 4

Evaluation

This chapter presents the evaluation of the proposed steganographic framework based on the Pix2Pix GAN, specifically focusing on the results obtained in Experiment 6, which demonstrated the most favorable outcomes among all experimental configurations.

It is important to note that the images displayed in Figure 3.12 were selectively chosen to illustrate successful instances. While the method was tested across a variety of secret images, it did not consistently achieve satisfactory performance for all inputs. Consequently, the evaluation presented here is limited to a subset of test cases in which the model performed as intended.

The assessment is structured around four key criteria commonly used in steganographic systems: *Steganographic Security*, *Hiding Capacity*, *Reconstruction Accuracy*, and *Robustness*. Additionally we also carry out a human evaluation to determine the robustness of the scheme to visual inspection. Each of these aspects is analyzed in the following sections.

4.1 Steganographic Security

Table 4.1 presents a comparative evaluation of several state-of-the-art steganographic methods using the Natural Image Quality Evaluator (NIQE) metric, which assesses perceptual image quality without requiring reference images. Lower NIQE

Method	**NIQE Score (± std)**
Baluja (Baluja 2017)	3.43 ± 0.08
HiNet (Jing et al. 2021)	2.94 ± 0.02
RIS (Y. Xu, Mou, et al. 2022)	3.13 ± 0.05
CRoSS (Yu et al. 2024)	3.04
HIS (Y. Xu, X. Zhang, et al. 2024)	3.10
Pix2Pix Diffusion (Ours)	4.13

Table 4.1: Comparison of NIQE scores

scores correspond to higher visual quality and more natural image appearance. *CRoSS* (Yu et al. 2024) and *HIS* (Y. Xu, X. Zhang, et al. 2024) demonstrate competitive performance. In contrast, the proposed *Pix2Pix Diffusion* model yields a higher NIQE score of 4.13, suggesting a relative degradation in image realism. This result implies a trade-off between visual quality and embedding capacity.

4.2 Hiding Capacity

Method	**Absolute Cap. (BPI)**	**Image Size**	**Relative Cap.(BPP)**
CamGAN (X. Liu et al. 2020)	256×256×8×3	256×256	24
CamGAN BW (Dharmawimala 2023)	256×256×8×2	256×256	16
CamGAN RGB (Lakshan 2024)	256×256×8×2×3	256×256	48
CRoSS (Yu et al. 2024)	512×512×8×3	512×512	24
Pix2Pix Diffusion (Ours)	512×512×8×3×2	512×512	48

Table 4.2: Comparison of hiding capacity

Table 4.2 presents a comparison of the hiding capacities of various steganographic methods in terms of absolute capacity (Bits Per Image, BPI) and relative capacity (Bits Per Pixel, BPP). Among the listed methods, *Pix2Pix Diffusion* demonstrates the highest hiding capacity, achieving 48 BPP with an image size of 512×512. This is on par with *CamGAN RGB*, which also reaches 48 BPP but operates on smaller 256×256 images, suggesting a higher spatial information density but limited resolution. The original *CamGAN* and *CRoSS* methods each achieve 24 BPP, though the latter works on higher-resolution images (512×512). *CamGAN BW*, using only two color channels instead of three, results in a reduced capacity of 16 BPP. The comparison indicates that the proposed Pix2Pix Diffusion approach significantly enhances the data embedding capability while preserving image resolution.

4.3 Reconstruction Accuracy

We evaluate the reconstruction accuracy of our proposed steganographic scheme in comparison to existing methods, including both closely related approaches and those that served as inspiration.

Table 4.3: Evaluation metrics comparison between different models

Method	**MSE**	**PSNR**	**SSIM**
Baluja (Baluja 2017)	-	27.51	0.89
CamGAN (X. Liu et al. 2020)	-	36.23	0.97
CamGAN RGB (Lakshan 2024)	546.25	21.39	0.66
CRoSS (Yu et al. 2024)	624.43	23.79	0.71
HIS (Y. Xu, X. Zhang, et al. 2024)	-	28.39	-
Pix2Pix Diffusion (Ours)	1614.80	16.98	0.68

Our method, *Pix2Pix Diffusion*, shows a lower PSNR and higher MSE compared to other approaches, which is expected due to the stochastic nature of diffusion models and their emphasis on perceptual realism over pixel-perfect reconstructions. Despite this, the SSIM score of 0.68 remains competitive, surpassing methods like *CamGAN RGB*, indicating that structural similarity is reasonably preserved. These results suggest that while our model may not excel in traditional distortion-based metrics, it balances reconstruction quality with robustness and perceptual plausibility.

4.4 Robustness

To assess the robustness of the steganographic scheme, we simulated JPEG compression loss and Gaussian Noise addition on a batch of container images and evaluated its impact on the recovery of the two secret images in each container. The two secret images were extracted from the corrupted container image, and the quality of the recovered images was quantified using four metrics: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), Learned Perceptual Image Patch Similarity (LPIPS) and Mean Square Error (MSE). These metrics were averaged over all secret images for each test scenario.

Noise σ	**LPIPS**	**PSNR**	**SSIM**	**MSE**
0	0.26	16.98	0.68	1614.80
5	0.30	16.78	0.64	1639.98
10	0.35	16.45	0.55	1768.54
15	0.40	16.27	0.48	1823.32
20	0.42	16.09	0.42	1902.70
25	0.48	15.64	0.37	2026.98
35	0.52	15.13	0.29	2188.28
50	0.58	15.07	0.26	2253.49
65	0.64	14.62	0.21	2450.32
80	0.66	14.19	0.17	2793.37
100	0.69	13.95	0.16	2901.12

Table 4.4: Recoverability at Different Noise Levels

A comparative analysis between the two types of degradation indicates:

Quality	LPIPS	PSNR	SSIM	MSE
100	0.28	17.00	0.69	1588.86
90	0.28	16.95	0.68	1628.12
80	0.27	16.99	0.68	1587.84
70	0.29	16.92	0.68	1627.55
60	0.30	16.79	0.68	1652.07
50	0.31	16.69	0.67	1690.04
40	0.32	16.44	0.67	1734.34
30	0.32	16.63	0.67	1699.41
20	0.35	16.33	0.66	1801.49
10	0.43	15.90	0.64	1923.74
5	0.46	15.28	0.62	2213.59

Table 4.5: Recoverability at Different Quality Levels

- **Resilience to JPEG Compression:** The scheme demonstrates stability under compression, with moderate changes in PSNR, SSIM, LPIPS, and MSE. The predictable impact of compression suggests that the recovery mechanism is adept at handling artifacts introduced by JPEG compression.
- **Vulnerability to Gaussian Noise:** The recovery performance shows a drastic deterioration when Gaussian noise is applied. The significant drop in both PSNR and SSIM, along with high LPIPS and MSE values, implies that noise introduces chaotic perturbations that the scheme is ill-equipped to mitigate.

While the steganographic scheme maintains acceptable performance under JPEG compression, it is notably more sensitive to Gaussian noise. For real-world applications in noisy environments it may be necessary to integrate noise-robust techniques or additional processing steps to enhance the quality of the recovered images.

Compared to existing methods, Our method demonstrates consistently robust performance under both Gaussian noise and JPEG compression, even though its

PSNR values are not always the highest. While *RIIS* achieves the best absolute scores, particularly under compression, our method maintains stable quality across all distortion levels, with less performance drop as degradations increase. Notably, unlike methods such as *HiNet* and *Baluja*, which show significant degradation under higher noise or compression, our approach provides a balanced trade-off between robustness and visual quality, making it better suited for real-world scenarios where input degradations are unpredictable and varied.

Method	**Gaussian Noise (σ)**			**JPEG Compression**		
	10	20	30	20	40	80
Baluja (Baluja 2017)	10.30	7.54	6.92	6.59	8.33	11.92
HiNet (Jing et al. 2021)	42.98	12.91	11.54	6.87	7.03	9.78
RIIS (Y. Xu, Mou, et al. 2022)	43.78	26.03	18.89	13.92	22.03	25.41
CRoSS (Yu et al. 2024)	23.79	21.89	20.19	21.02	21.74	22.74
HIS (Y. Xu, X. Zhang, et al. 2024)	23.49	21.88	20.02	23.21	26.11	26.23
Pix2Pix Diffusion (Ours)	16.45	16.09	15.42	16.33	16.44	16.99

Table 4.6: PSNR(dB) results comparison under Gaussian noise and JPEG compression

4.5 Human Evaluation

To supplement the quantitative metrics, a human evaluation study was conducted to assess the perceptual quality of container images and the semantic fidelity of recovered secret images along with the steganographic imperceptibility. Participants were presented with sets comprising the original secret image pairs, its corresponding container image, and the reconstructed outputs, and asked to provide qualitative ratings.

The evaluation was administered via a Google Form circulated through social media platforms [1]. Each question was repeated across five different sets to ensure

[1] `https://docs.google.com/forms/d/e/1FAIpQLSdcSecARdqO6IKBZfZkZPb_`

consistency. The final scores were computed by averaging participant responses across all instances. We were able to obtain 50 responses from participants who were mainly university students with no prior knowledge of digital stegenography.

Participants rated each sample on the following criteria:

- **Visual Realism** – How suspicious does the container image appear compared to others? (We provided other generated non-container images for comparison) (0: Clearly suspicious – 5: Indistinguishable)
- **Semantic Accuracy** – How clearly can the main content of the recovered secret images be identified? (0: Not identifiable – 5: Clearly identifiable)
- **Steganographic Imperceptibility** – Are there visible traces of the secret images in the container image? (0: Clearly visible – 5: No perceptible traces)

For the Pix2Pix-Diffusion model, the average scores across all evaluated samples were:

- **Visual Realism:** 3.8
- **Semantic Accuracy:** 4.1
- **Steganographic Imperceptibility:** 4.3

These results suggest that although traditional metrics such as PSNR may report relatively low scores, human observers consistently perceived the reconstructed images as semantically accurate. Moreover, the concealment was largely successful, with minimal traces of the secrets detected in the container images—even under side-by-side comparison with the secrets.

`cQUKhey9kzlZNFDSmLrCfABA3w/viewform?usp=preview`

Chapter 5

Conclusion

5.1 Conclusion to Research Questions

This research posed two primary questions regarding the feasibility and effectiveness of a hybrid GAN-diffusion steganographic framework:

1. How can a hybrid generative architecture combining GANs and diffusion models be used to construct a multi-image coverless steganographic pipeline?

   The experiments demonstrated that a hybrid architecture integrating GAN-based modules with pretrained diffusion models can indeed enable a coverless steganographic framework. The final proposed pipeline leveraged textual prompts as keys to guide image generation and reconstruction through diffusion, while downstream GANs refined the output for higher fidelity. Among all variants, the Pix2Pix + Diffusion approach showed the most reliable recovery of multiple RGB secret images, substantiating the viability of this fusion paradigm.

2. How does the use of diffusion-enhanced image synthesis affect the visual quality and imperceptibility of stego images compared to GAN-only methods?

   Empirical results showed that diffusion-enhanced synthesis improves the visual fidelity of container images over GAN-only systems. The hybrid approach increased hiding capacity (2x) and offered a much safer steganographic system than other cover modification techniques. However, prompt sensitivity and domain similarity remain challenges for consistent recovery.

These findings confirm that the combination of GAN and diffusion processes offers a novel and effective direction for coverless multi-image steganography.

5.2 Contributions

This thesis makes the following key contributions to the field of image steganography:

- A novel hybrid framework: Introduced a steganographic architecture that integrates GANs and diffusion models, enabling coverless hiding and recovery of multiple images—a first of its kind to explicitly fuse these paradigms in a unified pipeline.
- Systematic evaluation of multiple architectures: Designed and implemented four major approaches—VAE-based fusion, ResNet-based GAN, ESRGAN refinement, and a Pix2Pix-driven method.
- Robustness and quality benchmarking: Employed a rigorous set of metrics (NIQE, PSNR, SSIM, LPIPS) under various noise and compression scenarios to quantitatively validate the framework's performance.

Collectively, these contributions provide a secure foundation for future research in multi-image, coverless steganography using generative models.

5.3 Future Work and Limitations

5.3.1 Limitations

While the proposed framework demonstrates promising results in terms of imperceptibility and multi-image hiding capacity, it is not without limitations. One key constraint lies in the sensitivity to prompt or conditioning input selection, particularly in generative or diffusion-based approaches. If such prompts are not carefully chosen, the model may fail to generate container images that preserve the steganographic integrity of the hidden information.

Another notable limitation arises when the two secret images to be hidden belong to the same domain or exhibit substantial visual similarity. In such cases, the reverse diffusion process encounters difficulty in maintaining distinct representations for each image. Overlapping features can cause entanglement in the latent space, thereby degrading the fidelity of the recovered images and limiting the system's effectiveness in scenarios involving semantically or visually similar content.

Additionally, the versatility of the system is limited. The Pix2Pix GAN architecture, while effective within a specific domain, lacks generalizability across varied image contexts. To adapt the framework to new domains, separate models had to be trained from scratch, increasing computational overhead and limiting scalability. This domain dependency restricts the practicality of deploying the system in real-world, multi-domain environments without extensive retraining.

Another limitation of our framework pertains to steganographic security in relation to Kerckhoff's principle. Our approach assumes the confidentiality of model weights, which contravenes the principle's core tenet that the security of a system should not depend on the secrecy of its design or implementation, but solely on the secrecy of the key. While X. Hu et al. (2024) propose a methodology that circumvents this limitation by designing a system that remains secure even with publicly known model parameters, we did not adopt this approach due to the significant increase in complexity it entails and given that it was tailored for single image diffusion steganography.

Furthermore, encryption-based concealment of secret images was not feasible within our system architecture. Employing stream ciphers for image encryption tends to preserve localized patterns due to similar ciphertext values for adjacent or similar pixel groups, which can be visually perceptible. Conversely, block cipher encryption disrupts the statistical properties of the image to such an extent that it becomes incompatible with the invertible nature of diffusion models. Since our method relies on reconstructing the container image from an encoded latent or fused representation, such statistical destruction renders the reverse diffusion process inoperable.

5.3.2 Future Work

Building upon the current framework, an avenue for future research involves the integration of flow models to guide the denoising process within diffusion models. By parameterizing the forward process with normalizing flows, it could potentially lead to better secret recovery (Fischer et al. 2023).

Exploring the integration of flow models into the noising process presents a promising direction for enhancing the capabilities of diffusion-based steganographic frameworks.

Bibliography

Baluja, Shumeet (2017). "Hiding images in plain sight: Deep steganography". In: *Advances in neural information processing systems* 30.

Dharmawimala, Yashithi (2023). *Coverless Multi-Image Steganography using Generative Adversarial Networks.*

Fischer, Johannes S et al. (2023). "Boosting latent diffusion with flow matching". In: *arXiv preprint arXiv:2312.07360.*

Goodfellow, Ian J et al. (2014). "Generative adversarial nets". In: *Advances in neural information processing systems* 27.

Holub, Vojtěch, Jessica Fridrich, and Tomáš Denemark (2014). "Universal distortion function for steganography in an arbitrary domain". In: *EURASIP Journal on Information Security* 2014, pp. 1–13.

Howard, Andrew et al. (2019). "Searching for mobilenetv3". In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1314–1324.

Hu, Donghui et al. (2018). "A novel image steganography method via deep convolutional generative adversarial networks". In: *IEEE access* 6, pp. 38303–38314.

Hu, Xiaoxiao et al. (2024). "Establishing Robust Generative Image Steganography via Popular Stable Diffusion". In: *IEEE Transactions on Information Forensics and Security* 19, pp. 8094–8108. DOI: `10.1109/TIFS.2024.3444311`.

Isola, Phillip et al. (2017). "Image-to-image translation with conditional adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134.

Jing, Junpeng et al. (2021). “Hinet: Deep image hiding by invertible network”. In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4733–4742.

Jois, Tushar M, Gabrielle Beck, and Gabriel Kaptchuk (2023). “Pulsar: Secure Steganography through Diffusion Models”. In: *Cryptology ePrint Archive*.

Ke, Yan et al. (2019). “Generative steganography with Kerckhoffs’ principle”. In: *Multimedia Tools and Applications* 78.10, pp. 13805–13818.

Lakshan, K.D.M (2024). *GAN-based Coverless Multi-Image Steganography with RGB Secret Images*.

Liu, Jia et al. (2020). “Recent advances of image steganography with generative adversarial networks”. In: *IEEE Access* 8, pp. 60575–60597.

Liu, Lianshan et al. (2022). “A Larger Capacity Data Hiding Scheme Based on DNN”. In: *Wireless Communications and Mobile Computing* 2022.1, p. 5425674.

Liu, Ming-ming et al. (2017). “Coverless information hiding based on generative adversarial networks”. In: *arXiv preprint arXiv:1712.06951*.

Liu, Xiyao et al. (2020). “Camouflage generative adversarial network: Coverless full-image-to-image hiding”. In: *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, pp. 166–172.

Min-ha-zul Abedin, Md and Mohammad Abu Yousuf (2023). “StegoPix2Pix: Image Steganography Method via Pix2Pix Networks”. In: *Proceedings of the Fourth International Conference on Trends in Computational and Cognitive Engineering: TCCE 2022*. Springer, pp. 343–356.

Mode collapse — Wikipedia, The Free Encyclopedia (2024). Accessed: 2025-04-20. URL: `https://en.wikipedia.org/wiki/Mode_collapse`.

Rombach, Robin et al. (June 2022). “High-Resolution Image Synthesis With Latent Diffusion Models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695.

Shi, Haichao et al. (2018). “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. Springer, pp. 534–544.

Volkhonskiy, Denis, Boris Borisenko, and Evgeny Burnaev (2016). "Generative adversarial networks for image steganography". In.

Wang, Xintao et al. (2018). "Esrgan: Enhanced super-resolution generative adversarial networks". In: *Proceedings of the European conference on computer vision (ECCV) workshops*, pp. 0–0.

Xu, Sihan et al. (2024). "Cyclenet: Rethinking cycle consistency in text-guided diffusion for image manipulation". In: *Advances in Neural Information Processing Systems* 36.

Xu, Youmin, Chong Mou, et al. (2022). "Robust invertible image steganography". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 7875–7884.

Xu, Youmin, Xuanyu Zhang, et al. (2024). "Diffusion-Based Hierarchical Image Steganography". In: *arXiv preprint arXiv:2405.11523*.

Yang, Jianhua, Kai Liu, et al. (2018). "Spatial image steganography based on generative adversarial network". In: *arXiv preprint arXiv:1804.07939*.

Yang, Jianhua, Danyang Ruan, et al. (2019). "An embedding cost learning framework using GAN". In: *IEEE Transactions on Information Forensics and Security* 15, pp. 839–851.

Yu, Jiwen et al. (2024). "Cross: Diffusion model makes controllable, robust and secure image steganography". In: *Advances in Neural Information Processing Systems* 36.

Zhang, Kevin Alex et al. (2019). "SteganoGAN: High capacity image steganography with GANs". In: *arXiv preprint arXiv:1901.03892*.

Zhang, Zhuo et al. (2019). "Generative reversible data hiding by image-to-image translation via GANs". In: *Security and Communication Networks* 2019, pp. 1–10.