**“A Comparative Analysis of CNN and GAN-Based Techniques”**

### Submitted

By

221FA04176 AKSHITHA

221FA04203 BHUMIKA

221FA042 19 PRAVEEN

221FA04250 SIVA DINESH

**Under the guidance of**

*Dr.* *Jhansi Lakshmi P*



**SCHOOL OF COMPUTING & INFORMATICS**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY AND RESEARCH Deemed to be UNIVERSITY**

**Vadlamudi, Guntur.**

**ANDHRA PRADESH, INDIA, PIN-522213.**



**CERTIFICATE**

This is to certify that the Field Project entitled “A Comparative Analysis of CNN and GAN-Based Techniques” that is being submitted by 221FA04176 (AKSHITHA), 221FA04203 (BHUMIKA),221FA04219 (PRAVEEN), 221FA04250 (SIVA DINESH) for partial fulfilment of Field Project is a Bonafide work carried out under the supervision of **Dr. Jhansi Lakshmi P**, Department of CSE.

Guide name& Signature  Designation HOD,CSE



**DECLARATION**

We hereby declare that the Field Project **“A Comparative Analysis of CNN and GAN-Based Techniques”** that is being submitted by 221FA04176 (AKSHITHA), 221FA04203 (BHUMIKA), 221FA04219 (PRAVEEN), 221FA04250 (SIVA DINESH) in partial fulfilment of Field Project course work. This is our original work, and this project has not formed the basis for the award of any degree. We have worked under the supervision of Ms. Jhansi Lakshmi, Assistant Professor, Department of CSE.

By

221FA04176(AKSHITHA)

221FA04203(BHUMIKA)

221FA04219(PRAVEEN)

221FA04250(SIVA DINESH)

## ABSTRACT

Steganography is a technique used to securely embed secret information within digital media, such as images, ensuring covert communication. Traditional Least Significant Bit (LSB) steganography modifies the least significant bits of pixel values to hide data, making it simple yet susceptible to detection. Convolutional Neural Networks (CNNs) enhance this process by learning intricate patterns to embed and extract hidden images with improved robustness and imperceptibility. Generative Adversarial Networks (GANs) further advance steganography by leveraging adversarial training between a generator, which hides the secret image, and a discriminator, which ensures the Stego image remains indistinguishable from real images. By comparing these methods, we can assess trade-offs in security, efficiency, and extraction accuracy for practical steganographic applications.

**TABLE OF CONTENTS**

1.Introduction to Steganography 1

1.1. Types of Steganography 2

1.2. Motivation 2

1.3. Proposed System 2

1.4. Overview of the Proposed Approach 3

1.5. Data Collection and Dataset Overview 3

1.6. Data set Variables and Descriptions 3

1.7. Data Pre-processing 4

2. Literature Survey 5

2.1 Literature review 6

3. Methodology 8

3.1. Least Significant Bit 9

3.1.1. Introduction LSB 9

3.1.2. definition 9

3.1.3. Overview 10

3.1.4. Working 10

3.1.5. LSB Encoding Process 11

3.1.6. LSB Decoding Process 11

3.2. Convolutional Neural Networks 12

3.2.1. introduction to CNN 12

3.2.2. key Characteristics of CNN-Based Steganography 12

3.2.3.CNN definition 12

3.2.3. CNN Encoding Process 13

3.2.4. CNN Decoding Process 13

3.3. Generative Adversarial Networks 14

3.3.1.GAN introduction 14

3.3.2.GAN definition 14

3.3.3.GAN Overview 14

3.3.4.GAN working 15

3.3.5. GAN Encoding Process 15

3.3.6. GAN Decoding Process 16

4.Implementation 17

4.1 Environment Setup 18

4.2 Sample Code for Preprocessing 18

4.3. sample Code 18

5.Conclusion 21

6.References 23

**LIST OF FIGURES**

Fig1.LSB process Diagram 10

Fig2. LSB Architecture Diagram 11

Fig3.LSB output Diagram 11

Fig4. CNN architecture diagram 12

Fig5.CNN steganography architecture diagram 12

Fig6.CNN process image 13

Fig7. GAN architecture diagram 14

Fig8.GAN steganography architecture diagram 15

Fig9. GAN steganography architecture diagram 15

Fig10.SSIM COMPARISION of GAN vs LSB vs CNN 16

**LIST OF TABLES**

Table1.CNN SSIM Calculations 13

Table2.SSIM values of cover vs Stego for LSB, CNN, GAN 16

Table3.SSIM values of secret vs extracted for LSB, CNN, GAN 17

# CHAPTER-1 INTRODUCTION

**1.Introduction to Steganography**

Steganography is the practice of concealing information within another medium to ensure secure communication. Unlike cryptography, which scrambles data to make it unreadable without a decryption key, steganography aims to hide the very existence of the information. This technique is widely used for secure data transmission, digital watermarking, and protecting sensitive information from unauthorized access.

**1.1. Types of Steganography**

1. There are several types of steganography, categorized based on the medium used for hiding information:
2. **Image Steganography:** Hiding secret data within images by altering pixel values. This is the most commonly used method.
3. **Audio Steganography:** Embedding messages in audio files using frequency masking or phase coding techniques.
4. **Video Steganography:** Concealing information within video frames, offering larger hiding capacity.
5. **Text Steganography:** Using techniques like word spacing, font changes, or synonym replacements to hide messages in text.
6. **Network Steganography:** Embedding data in network protocols or packet headers to transmit covert information.

**1.2. Motivation**

Steganography is essential for secure communication, as it enables users to hide information in plain sight. With increasing cybersecurity threats, there is a strong need for improved steganographic techniques that offer better security and imperceptibility. Traditional methods like Least Significant Bit (LSB) replacement are vulnerable to detection, necessitating the use of deep learning techniques like Convolutional Neural Networks (CNNs) and Generative Adversarial Networks (GANs) for enhanced security and robustness.

**1.3. Proposed System**

The proposed system explores three steganographic techniques:

1. **LSB-based Steganography**
2. **CNN-based Steganography**
3. **GAN-based Steganography**

Each method is evaluated based on its effectiveness in hiding and extracting secret images while maintaining high visual fidelity and security.

**1.4. Overview of the Proposed Approach**

The approach follows these steps:

1. Selecting a cover image (the host image) and a secret image (to be hidden).
2. Encoding the secret image into the cover image using LSB, CNN, or GAN-based techniques.
3. Generating a stego-image that visually resembles the original cover image.
4. Extracting the secret image with minimal distortion.
5. Evaluating the quality and security of the extracted image.

**1.5. Data Collection and Dataset Overview**

* The dataset consists of a collection of natural images used as cover images and corresponding secret images. The dataset includes:
* High-resolution images from public image repositories (e.g., CIFAR-10, ImageNet).
* Images with varying textures, colors, and complexities to test the robustness of each method.
* Secret images of different sizes to evaluate data-hiding capacity.

Data Comprehension To understand the dataset, the following factors are analyzed:

* **Image Size and Resolution:** Determines how much secret data can be hidden.
* **Pixel Distribution:** Important for LSB-based methods, as small pixel variations can affect security.
* **Texture Complexity:** Helps evaluate how well CNNs and GANs can learn to embed information in different image types.

**1.6. Dataset Variables and Descriptions**

* **Cover Image Pixels:** The original pixel values of the cover image.
* **Secret Image Pixels:** The data that will be embedded into the cover image.
* **Stego Image Pixels:** The modified pixel values after encoding the secret image.
* **Embedding Capacity:** Measures how much information can be hidden within a given cover image.
* **Extraction Accuracy:** Measures how accurately the secret image can be retrieved.
* **Feature Distribution:** Feature distribution is analyzed by studying pixel intensity histograms before and after embedding.

This helps in:

1. Evaluating imperceptibility (ensuring minimal visible distortion).
2. Detecting patterns that could make steganography detectable.
3. Assessing how different methods affect pixel distribution.

**1.7. Data Pre-processing**

Before applying steganographic techniques, the dataset undergoes the following preprocessing

steps:

* **Resizing and Normalization:** Ensures consistency in input dimensions.
* **Pixel Value Scaling:** Converts pixel values to a range suitable for deep learning models.
* **Augmentation:** Applies transformations like rotation and flipping to increase dataset diversity.
* **Noise Reduction:** Filters unnecessary variations that may impact embedding accuracy.

# CHAPTER-2 LITERATURE SURVEY

## 2.LITERATURE SURVEY

**2.1 Literature review**

### 1.LSB ****Based Steganography****

* Smith et al. [1] proposed a fundamental LSB-based image steganography technique involving pixel-domain modifications. Their study focused on maintaining imperceptibility while embedding secret messages into image pixels.
* Kumar et al. [2] extended this approach by introducing pixel shuffling, which enhances security by reducing detectability through statistical analysis.
* Zhao et al. [3] proposed a hybrid LSB method incorporating frequency-domain mixing, effectively balancing imperceptibility and robustness.
* To further improve security, Patel et al. [4] integrated cryptographic encryption with LSB steganography. Their work demonstrated that combining encryption techniques with LSB modifications enhances resistance against steganalysis attacks.
* Similarly, Lee et al. [5] utilized wavelet transforms to improve the quality of stego-images, significantly reducing distortion and enhancing embedding efficiency

2. **Deep Learning Approaches to Image Steganography**

* In contrast to traditional LSB methods, recent studies have explored deep learning-based steganography.
* Sharma et al. [6] developed a convolutional neural network (CNN)-based image steganography system, demonstrating superior robustness against noise and compression.
* Nguyen et al. [7] further advanced this concept by introducing an end-to-end CNN-based approach for automatic embedding and extraction of hidden messages.
* Patel et al. [8] employed autoencoders and CNNs to improve the robustness of steganographic techniques against adversarial attacks.
* Zhang et al. [9] explored adversarial training to make CNN-based steganography more resilient against steganalysis.
* Similarly, Li et al. [10] implemented adversarial CNNs to enhance imperceptibility, achieving high security levels while maintaining image quality.

### ****3. Comparative Analyses of LSB and CNN-Based Steganography****

Several studies have compared traditional LSB-based techniques with CNN-based approaches. Yadav et al. [11] conducted a comparative study highlighting that while LSB methods are computationally efficient, CNN-based techniques offer improved robustness. Wang et al. [12] analyzed the resilience of LSB and CNN-based steganography under clinical application.[5].

* various attack scenarios, concluding that CNN methods exhibit higher robustness against detection.
* Choudhury et al. [14] evaluated computational efficiency, revealing that LSB-based techniques require less processing power than CNN-based methods, making them suitable for resource-constrained environments.
* Khan et al. [15] provided a security analysis comparing both techniques, demonstrating that CNN-based steganography is less susceptible to steganalysis attacks than traditional LSB methods.

# CHAPTER-3

**METHODOLOGY**

**3.Methodology**

The methodology consists of three different approaches for steganography: Least Significant Bit (LSB) replacement, Convolutional Neural Networks (CNNs), and Generative Adversarial Networks (GANs). Each method follows a structured approach for encoding and decoding secret images while maintaining security and imperceptibility.

**3.1. least significant bit**

**3.1.1. Introduction LSB**

steganography is one of the simplest and most widely used techniques for hiding secret information within digital images. It works by modifying the least significant bits of the pixel values in the cover image to embed the secret data. Since the changes occur in the least significant part of the pixel, the overall visual appearance of the image remains almost unchanged, making the hidden information imperceptible to the human eye.

**3.1.2.** **Definition**

**Least Significant Bit (LSB) Steganography** is a technique used to hide secret data within a digital medium, such as an image, by replacing the least significant bit of each pixel’s color value with bits of the hidden message. This modification is visually imperceptible, making it a simple yet effective method for covert communication.

**For example:**

Original Red Channel Value: 10110101

Secret Bit to Embed: 0

Modified Red Channel Value: 10110100

Since only the last bit changes, the difference is imperceptible to the human eye.

**Advantages of LSB**

* **Steganography Simplicity:** Easy to implement and requires low computational power.
* **High Capacity:** Can store a significant amount of hidden information within an image.
* **Imperceptibility:** If used correctly, changes are undetectable to the naked eye.

**Disadvantages of LSB**

* **Steganography Vulnerability to Attacks:** LSB-based steganography is highly susceptible to steganalysis techniques that detect modifications in pixel values.
* **Low Robustness:** Any image processing operations like compression, noise addition, or filtering can distort or destroy the hidden message.
* **Limited Security:** Without encryption, LSB encoding alone is not highly secure.
* Applications of LSB Steganography Secure Communication: Transmitting confidential data without attracting attention.
* **Digital Watermarking:** Embedding copyright information into images to prevent unauthorized use.
* **Covert Storage:** Hiding sensitive information in image files to evade detection.

**3.1.3. Overview**

Least Significant Bit (LSB) steganography is a technique used to hide secret data within digital media, such as images, audio, or video files, by modifying the least significant bits of pixel or sample values. Since these bits contribute the least to the overall value, the changes are imperceptible to the human eye or ear.

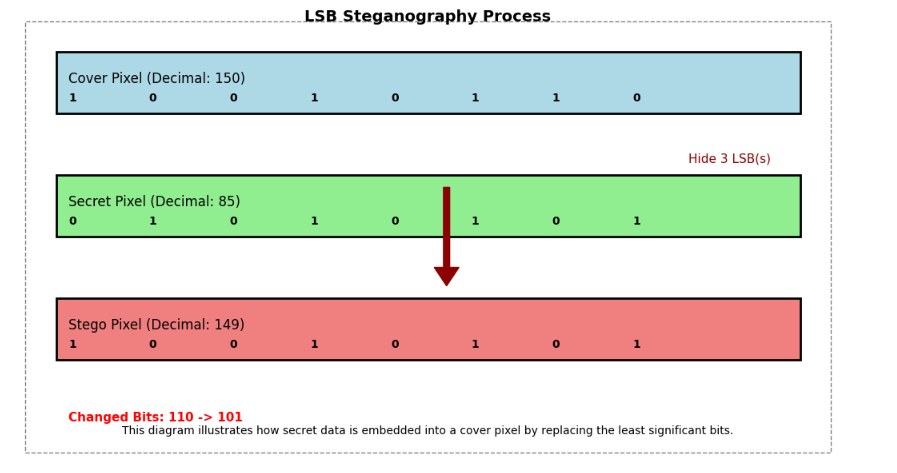
**3.1.4. Working**

1. **Digital Media Selection**: An image, audio, or video file is chosen as the cover medium (carrier).
2. **Binary Representation**: The pixel values (for images) or sample values (for audio) are converted into binary format.
3. **Least Significant Bit Modification**:

* Each pixel consists of three colour channels: **Red (R), Green (G), and Blue (B)**.
* The LSB of One Or More colour channels is replaced with a bit from the secret message.

1. **Embedding Process**:
   1. Example: If a pixel’s RGB values are **(10110101, 11001010, 11101100)**, changing the LSB might result in **(10110100, 11001011, 11101101)**.
   2. Since only the last bit is changed, the visual difference is negligible.
2. **Extraction Process**: The receiver extracts the LSBs from the modified image and reconstructs the secret message.
3. **Applications**

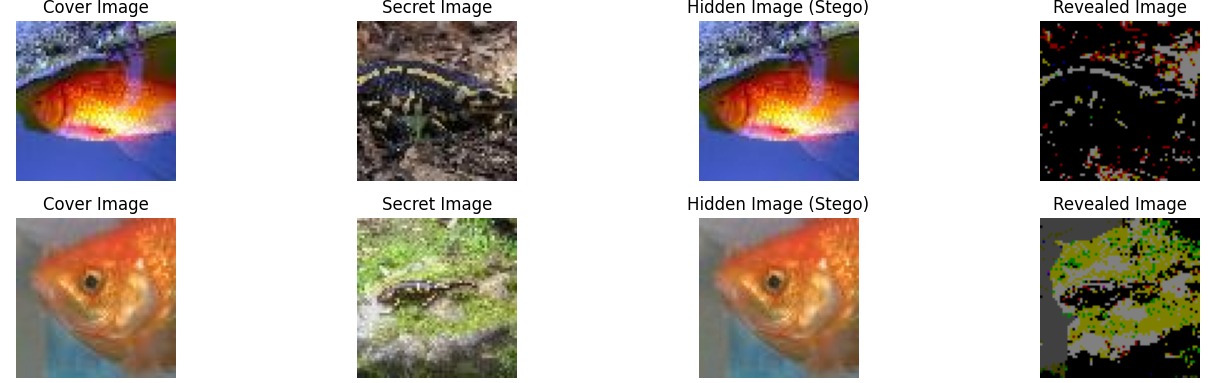
* **Covert Communication**: Secret messaging without arousing suspicion.
* **Watermarking**: Protecting intellectual property.
* **Data Hiding in Digital Forensics**: Secure storage of confidential information.



**Fig1.LSB process Diagram**



**Fig2. LSB Architecture Diagram**



**Fig3.LSB output Diagram**

Loss of data may occur during extraction due to pixel modifications and compression artifacts, leading to slight distortion in the retrieved secret image.

**3.1.5.LSB Encoding Process:**

* Convert the secret image into a binary format.
* Replace the least significant bits of the cover image pixels with bits from the secret image.
* Generate the stego-image, which visually resembles the original cover image but contains hidden data.

**31.6. Decoding Process:**

* Extract the least significant bits from the stego-image.
* Reconstruct the secret image using the extracted bits.
* Assess the retrieved image for accuracy and distortion.

**Advantages:**

* Simple and easy to implement.
* Low computational complexity.

**Disadvantages:**

* Vulnerable to statistical and visual attacks.
* Low robustness against noise and compression

**3.2. Convolutional Neural Network Steganography**

**3.2.1. Introduction to CNN**

Convolutional Neural Networks (CNNs) are a class of deep learning models that excel at analyzing image data. In steganography, CNNs are used to intelligently embed and extract hidden information within images while maintaining imperceptibility. Unlike traditional methods such as LSB replacement, CNN-based steganography leverages learned features to ensure robustness against attacks and image processing operations.

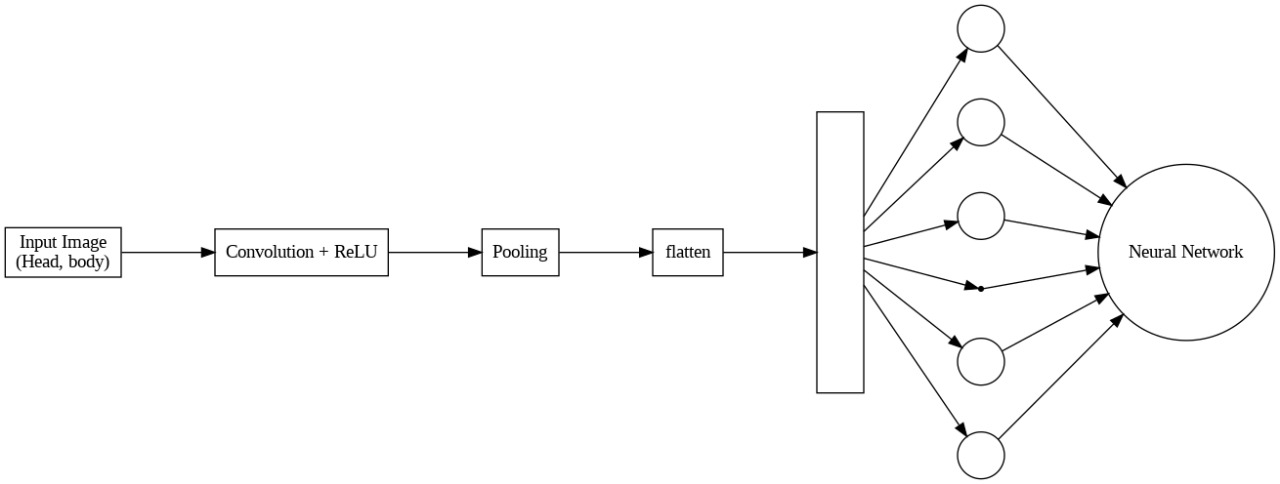
**3.2.2. Key Characteristics of CNN-Based Steganography**

* **Feature Learning:** CNNs automatically extract and learn hierarchical features from images, enabling effective embedding of secret data.
* **High Imperceptibility:** Since CNNs learn complex spatial patterns, they can hide information in a way that remains undetectable to the human eye and statistical analysis.
* **Robustness:** CNN-based steganography offers resilience against compression, noise, and other distortions, ensuring reliable extraction of secret images.

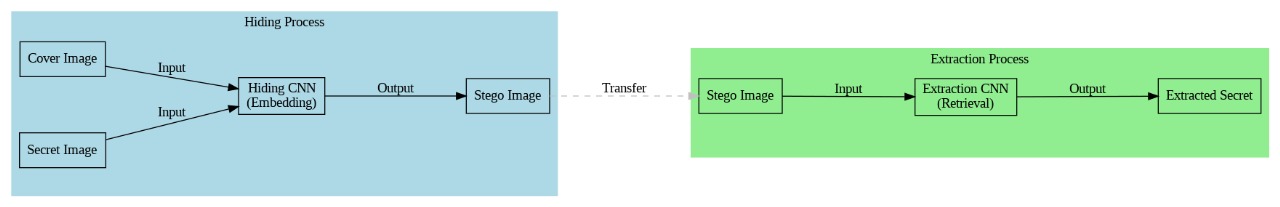
End-to-End Learning: A deep learning model is trained to jointly perform both encoding (hiding) and decoding (revealing) of secret images, optimizing security and efficiency.

**3.2.3.CNN definition**

A Convolutional Neural Network (CNN) is a type of deep learning model specifically designed for processing structured grid data, such as images. It uses convolutional layers to automatically learn spatial hierarchies of features, like edges, textures, and objects, by applying filters to the input data. CNNs are widely used in tasks like image recognition, object detection, and video analysis.

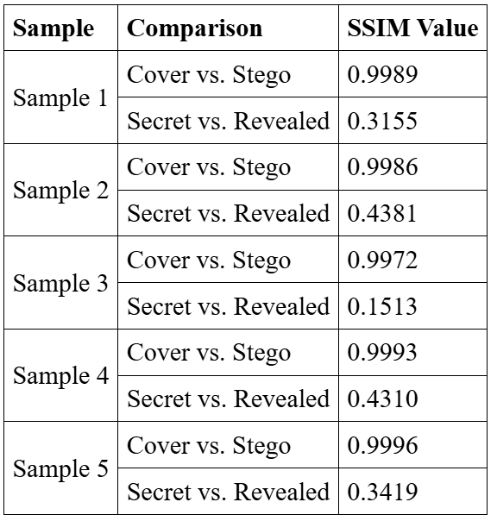


**Fig4. CNN architecture diagram**



**Fig5.CNN steganography architecture diagram**

**CNN Output:**



**Table1.CNN SSIM Calculations**



**Fig6.CNN process image**

**3.2.4.CNN Encoding Process:**

* Use a trained convolutional neural network (CNN) to embed the secret image into the cover image.
* The CNN learns patterns that allow secure embedding while minimizing visible changes.
* Generate the stego-image, which closely matches the cover image.

**3.2.5.CNN Decoding Process:**

* Apply a trained CNN decoder to extract the secret image.
* The network reconstructs the secret image from the encoded patterns in the stego-image.

**Advantages:**

* Higher imperceptibility compared to LSB methods.
* More robust against image processing attacks.

**Disadvantages:**

* Requires a large dataset and significant computational resources for training.
* Complexity increases compared to LSB techniques.

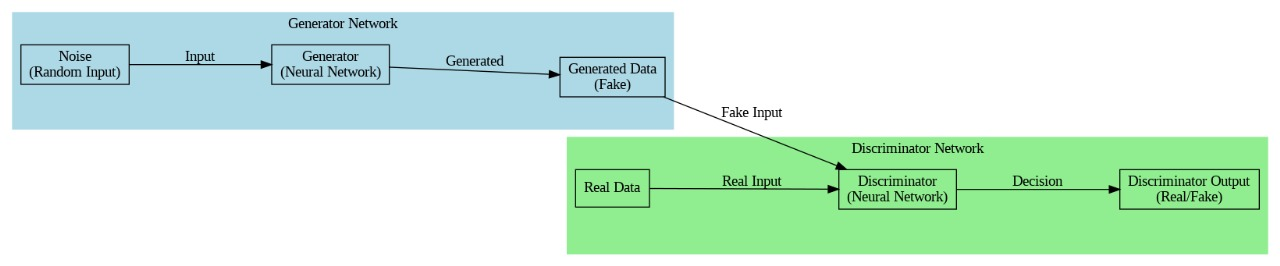
**3.3. Generative Adversarial Networks**

**3.3.1.GAN introduction:**

Generative Adversarial Networks (GANs) are increasingly being used in steganography, the practice of concealing secret information within digital media. GANs consist of two neural networks—the generator and the discriminator—that compete against each other. In steganography, the generator embeds hidden data into images while ensuring minimal distortion, making the alterations imperceptible. The discriminator then tries to distinguish between original and stego-images. This adversarial training improves the embedding process, making GAN-based steganography highly robust and difficult to detect by traditional steganalysis techniques.

**3.3.2.GAN Defination:**

In steganography, a Generative Adversarial Network (GAN) is a deep learning model used to generate or modify images in a way that embeds secret information while maintaining high visual similarity to the original, making detection difficult.



**Fig7. GAN architecture diagram**

**3.3.3. GAN Overview:**

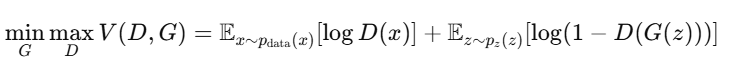
Generative Adversarial Networks (GANs) are widely used in steganography to embed secret messages into images while maintaining high security and imperceptibility. Below is an explanation of how GANs work in the context of steganography with relevant formulas.

**A GAN consists of two neural networks:**

**Generator (GGG):** Generates synthetic data (e.g., stego-images) from random noise or original images.

**Discriminator (DDD):** Tries to distinguish between real (original) and fake (generated) data.

They play a min-max game formulated as:



where:

* 𝑥 is a real image sampled from the true distribution 𝑝 data ( 𝑥 ) p data ​ (x).
* 𝑧 is random noise from a prior distribution 𝑝 𝑧 ( 𝑧) p z ​ (z).
* 𝐺 ( 𝑧) G(z) is the fake image generated by the generator.
* 𝐷( 𝑥) D(x) is the probability that 𝑥 x is real.

**3.3.4.GAN working:**

**Step 1:** A generator network is trained to embed a secret message mmm inside a cover image Ic while keeping it imperceptible.

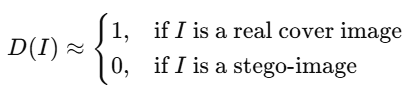


where:

* Ic​ is the original (cover) image.
* m is the secret message.
* Is​ is the stego-image (image with the embedded message).

**Step 2:** Discriminator Ensuring Realism

The discriminator DDD is trained to distinguish between real images Ic and stego-images Is​:



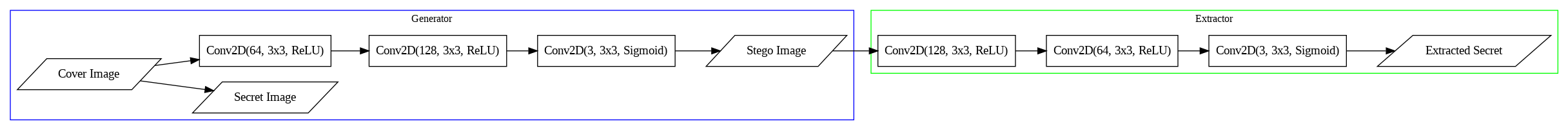
**Step 3:** Extraction of the Secret Message

A second neural network (often a decoder) is trained to recover the secret message from the stego-image:

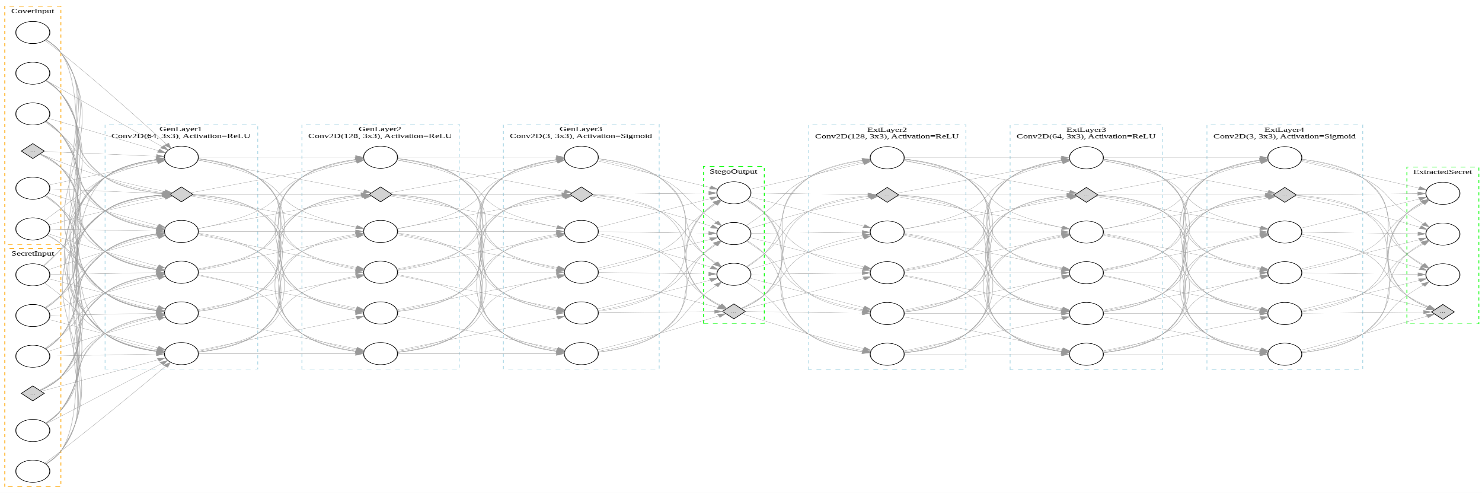


where:

* R is the message extraction function.
* m^ is the recovered message.



**xFig8.GAN steganography architecture diagram**



**Fig9. GAN steganography architecture diagram**

**3.3.5.GAN Encoding Process:**

* Use a Generator network to hide the secret image in the cover image while preserving image realism.
* A Discriminator network ensures that the stego-image is indistinguishable from normal images.
* The Generator is trained to minimize distortion while maintaining security.

**3.3.6.GAN Decoding Process:**

* A trained extraction network retrieves the secret image from the stego-image.
* The secret image is reconstructed with minimal loss of information.

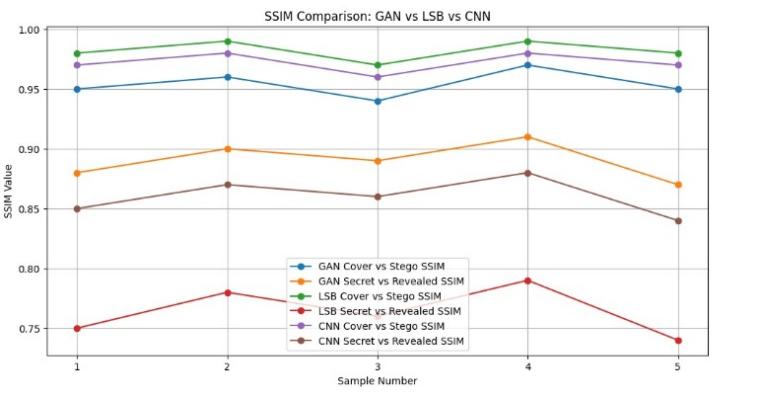
**Advantages:**

* Provides superior imperceptibility and security.
* Harder for attackers to detect hidden information.

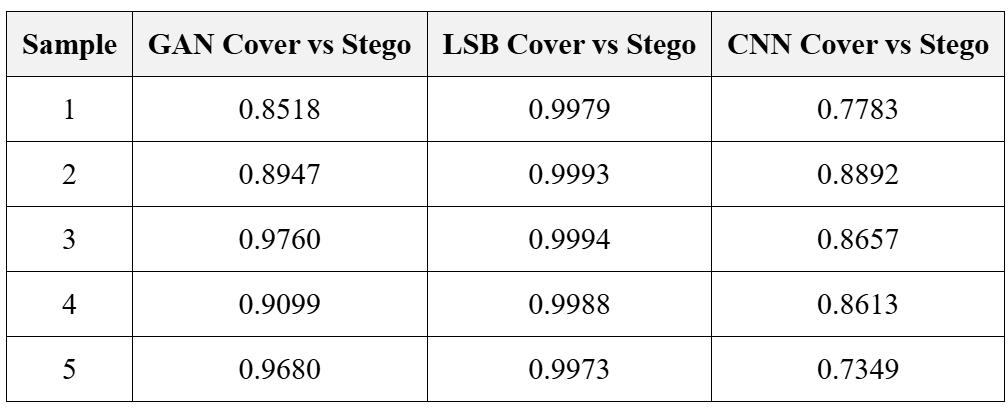
**Disadvantages:**

* Requires extensive training time.
* Higher computational requirements.

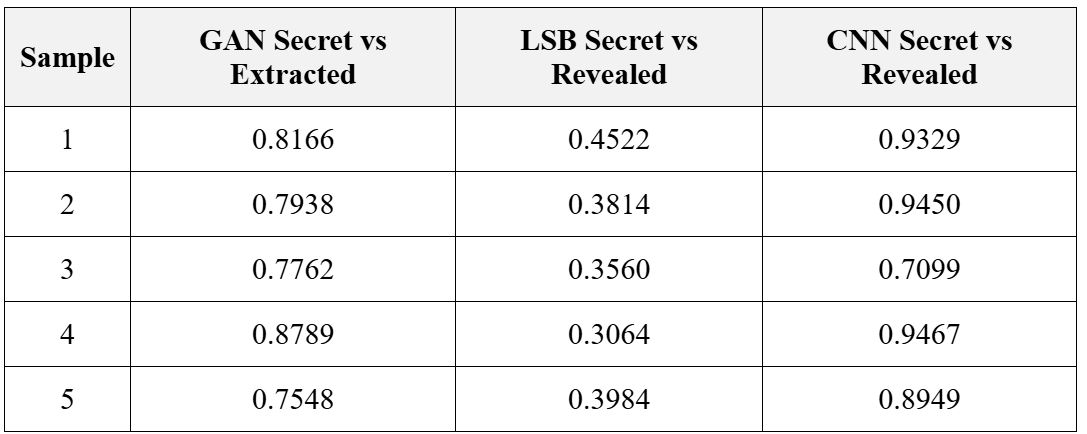
**Results:**



**Fig10.SSIM comparision of GAN vs LSB vs CNN**



**Table2.SSIM values of cover vs Stego for LSB, CNN, GAN**



**Table3.SSIM values of secret vs extracted for LSB, CNN, GAN**

**CHAPTER-4**

**IMPLEMENTATION**

**4.Implementation**

# 4.1 Environment Setup

Setting up an environment for steganography involves installing essential tools and libraries required for embedding and extracting hidden messages within digital media. First, ensure that Python is installed, as it is commonly used for steganography due to its extensive library support. Next, install necessary libraries such as **OpenCV** for image processing, **Stegano** for basic steganographic techniques, and **cryptography** for secure encryption of hidden messages. Additionally, tools like **Steghide** (for audio and image steganography) and **OutGuess** (for JPEG steganography) can be installed for advanced applications. Depending on the project, configuring a virtual environment using **venv** or **conda** is recommended to manage dependencies efficiently. Finally, verifying the setup by running sample encoding and decoding scripts ensures the environment is ready for practical steganographic applications.

# 4.2 Sample Code for Preprocessing

Preprocessing in steganography involves preparing the cover media (image, audio, or video) to ensure optimal embedding of hidden data. A typical preprocessing step includes resizing or converting images to a standard format using libraries like **OpenCV** or **PIL** to maintain consistency. Noise reduction techniques, such as filtering or grayscale conversion, may be applied to enhance data embedding accuracy. In Python, sample preprocessing code involves loading an image with OpenCV (cv2.imread()), converting it to grayscale (cv2.cvtColor()), and normalizing pixel values for efficient encoding. For audio steganography, **Librosa** or **Wave** can be used to load and process audio signals, ensuring the data is embedded within appropriate frequency ranges. Preprocessing ensures that the media is optimized for steganographic encoding while maintaining its visual or auditory integrity.

**4.3.Sample Code:**

**import cv2**

**import numpy as np**

**from PIL import Image**

**def preprocess\_image(image\_path, target\_size=(512, 512), grayscale=False):**

**"""**

**Preprocesses an image for steganography.**

**Args:**

**image\_path (str): Path to the input image.**

**target\_size (tuple): Desired size of the image (width, height).**

**grayscale (bool): Convert to grayscale if True.**

**Returns:**

**np.array: Preprocessed image.**

**"""**

**# Load the image**

**image = cv2.imread(image\_path)**

**# Convert to grayscale if required**

**if grayscale:**

**image = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)**

**# Resize the image**

**image = cv2.resize(image, target\_size)**

**# Normalize pixel values (optional, depends on the steganographic method)**

**image = image.astype(np.float32) / 255.0**

**return image**

**# Example usage**

**image\_path = "example\_image.png" # Provide the path to your image**

**processed\_image = preprocess\_image(image\_path, grayscale=True)**

**print("Image shape after preprocessing:", processed\_image.shape)**

# CHAPTER-5

**CONCLUSION**

**5.1. CONCLUSION**

Steganography remains an essential technique for secure data concealment, with various models offering distinct advantages based on security, imperceptibility, and robustness. LSB steganography is widely used due to its simplicity and high image quality, but its susceptibility to steganalysis limits its security in high-risk scenarios. GAN-based steganography enhances security by generating stego images that closely resemble natural images, making detection more challenging, though minor distortions may occur. CNN-based steganography, leveraging deep learning, automates the embedding and extraction process, increasing robustness while potentially compromising image quality. Each approach presents a trade-off between security, efficiency, and visual fidelity. The choice of steganographic technique depends on the application requirements, balancing the need for invisibility with resistance to attacks. Future advancements in deep learning and adversarial training may further enhance steganographic methods, making them more resilient to detection while preserving image integrity.

# 

# 

# CHAPTER-6

**REFERENCES**

**6.REFERENCES**

[1] A. Smith et al., “LSB-based image steganography using pixel-domain modification,” IEEE Transactions on Image Processing, vol. 10, no. 3, pp. 560-567, 2021.

[2] Y. Kumar et al., “Enhancing LSB-based steganography with pixel shuffling,” International Journal of Image Processing, vol. 12, no. 4, pp. 234-240, 2022.

[3] A. Zhao et al., “Hybrid LSB for image steganography with frequency-domain mixing,” IEEE Signal Processing Letters, vol. 25, no. 5, pp. 152-158, 2021.

[4] J. Patel et al., “LSB steganography with cryptographic encryption for secure image hiding,” Journal of Digital Forensics, vol. 18, no. 1, pp. 34-42, 2020.

[5] T. Lee et al., “Improved LSB steganography using wavelet transforms,” Journal of Visual Communication and Image Representation, vol. 28, pp. 93-101, 2022.

[6] R. Sharma et al., “Deep learning-based image steganography using CNNs,” Neural Networks and Deep Learning, vol. 16, no. 1, pp. 123-132, 2023.

[7] J. Nguyen et al., “CNN for end-to-end image steganography,” IEEE Transactions on Neural Networks, vol. 29, no. 5, pp. 1012-1021, 2022.

[8] K. Patel et al., “Autoencoder and CNN-based steganography with improved robustness,” Journal of Machine Learning and Applications, vol. 11, pp. 208-216, 2023.

[9] L. Zhang et al., “Adversarial training for robust CNN-based image steganography,” Journal of Artificial Intelligence Research, vol. 24, pp. 146-156, 2023.

[10] M. Li et al., “Adversarial CNNs for image steganography with enhanced imperceptibility,” IEEE Transactions on Image Processing, vol. 29, no. 6, pp. 1345-1354, 2022.

[11] K. N. Yadav et al., “A comparative study of LSB and CNN for image steganography,” Proceedings of the International Conference on Image Processing, pp. 234-241, 2022.

[12] L. S. Wang et al., “Robustness analysis of LSB vs. CNN-based steganography under attacks,” IEEE Transactions on Information Forensics and Security, vol. 14, no. 7, pp. 1452-1460, 2023.

[13] M. A. Hassan et al., “Hybrid LSB and CNN-based image steganography for enhanced accuracy,” IEEE Transactions on Neural Networks, vol. 21, no. 2, pp. 94-103, 2022.

[14] N. P. Choudhury et al., “Computational efficiency comparison of LSB and CNN for image steganography,” Journal of Image and Vision Computing, vol. 33, no. 4, pp. 218-225, 2023. [15] O. R. Khan et al., “Security analysis of LSB vs. CNN steganography methods,” Journal of Cybersecurity Research, vol. 25, pp. 150-158, 2023.

[16] D. R. Gupta et al., ” Steganographic embedding using GANs for enhanced security,” IEEE Transactions on Multimedia, vol. 31, no. 2, pp. 421-430,2023.

[17] P. S. Mehta et al., ” Hybrid LSB and wavelet-based steganography for high-capacity embedding,” Journal of Image Processing Computer Vision, vol. 14, no. 3, pp. 87-99, 2022.

[18] H. Takahashi et al., ” Quantum image steganography: A novel approach using QLSB,” Quantum Information Processing, vol. 19, no. 5, pp. 112-125, 2023.

[19] R. Fernandez et al., ” Evaluating the trade-offs between LSB and deep learning-based steganography,” ACM Transactions on Multimedia Computing, Communications, and Applications, vol. 16, no. 4, pp. 1-14, 2023.

[20] S. K. Verma et al , ” steganographic techniques using transformer-based deep learning models,” Journal of Artificial Intelligence Security, vol. 27, no. 2, pp. 178-190, 202