Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 31 No. 1s (2024)

CompressGAN: A Generative Adversarial Network-Based Framework for Learned Image Compression

Dharmesh Dhabliya, Dr. Priya Vij

Research Scholar, Department of Computer Science and Engineering

Kalinga University Raipur

Department of Computer Science and Engineering

Kalinga University Raipur

Article History:

Received: *12-02-2024*

Revised: *10-04-2024*

Accepted: *24-04-2024*

Abstract:

With the rise of high-resolution digital material, there is a much greater need for effective picture compression methods. Traditional image compression methods, like JPEG and PNG, have a hard time keeping the compression rate and quality of the picture in balance, especially when the bitrate is low. We present CompressGAN, a new method for learning to compress images that is based on Generative Adversarial Networks (GANs). An encoder-decoder network and a discriminator are paired in the suggested design so that it can learn small latent representations while keeping the quality of the images. CompressGAN is different from other methods because it uses hostile training to find features that are important to perception. This makes reconstructions that look good even at high compression levels. The decoder shrinks the picture data into a small hidden code. This code is then quantised and entropy-coded to make it easier to store. The decoder builds the picture back up from the compressed version using both pixel-wise reconstruction loss and hostile loss from the discriminator as guides. We add a perceived loss calculated from a feature extractor network that has already been trained to improve realism and lower artefacts. CompressGAN does better than traditional codecs and a few new learnt compression models in terms of PSNR, SSIM, and perceived quality measures, as shown by experiments done on standard picture datasets. Furthermore, our approach keeps conceptual continuity and visual features better than baselines, especially when bit rates are low. CompressGAN can be trained from start to finish and can be changed to work with different compression rates by changing the quantisation levels or latent space density. This framework looks like a good way to handle apps with limited storage space and bandwidth, like mobile images, monitoring systems, and online content delivery.

Keywords: Image Compression, Generative Adversarial Network, Deep Learning, Perceptual Quality, Learned Representations, Neural Codec

1. Introduction

Due to the widespread use of high-resolution imaging devices and the fast growth of online content sites, digital media has grown very quickly. This has increased the need for effective and high-quality picture compression methods. Image compression standards like JPEG, PNG, and WebP have been useful to the digital world for decades by making files smaller so they can be sent more quickly and take up less space. But these old codecs depend on changes and quantisation methods that are carefully made by hand, and they don't always work well at lower bit rates to keep the quality of the sound. The problems with old ways of doing things become clearer as digital photos get bigger and more complicated, and as they are used for things like real-time monitoring, social media, and taking pictures on the go that need to be fast and accurate. Neural network-based picture compression models were created because of the need for smart, data-driven solutions that can learn the best ways to reduce data from it [1].

With the help of autoencoders, variational autoencoders (VAEs), and convolutional neural networks (CNNs), recent advances in deep learning have made strong replacements to old compression methods. In terms of rate-distortion trade-offs, these models learn to pack pictures into small latent representations and restore them with little loss. They often do a better job than standard codecs. Even with these changes, one problem that learnt compression systems often have is that it's hard to keep the sound of what you hear, especially when compressing at very low bitrates [2]. One way to use the power of Generative Adversarial Networks (GANs) is in this way. Because they can model complex data patterns and make lifelike effects, GANs have done amazingly well in high-fidelity picture creation and super-resolution jobs. Their antagonistic training system pushes the generator to make outputs that can't be told apart from real images. This builds a strong base for reconstructing pictures in a way that is aware of how people see them. We present CompressGAN, a new GAN-based system for learnt picture compression that fills the gap between small representation and high quality perception. CompressGAN takes the best parts of an encoder-decoder design and the competitive nature of GANs to learn end-to-end picture compression models that can make reconstructions that look good. The encoder turns a picture input into a latent space with few dimensions. The latent space is then quantised and encoded with entropy to make it easier to store or send. To make sure the picture is accurate, the encoder rebuilds it from the hidden representation using both reconstruction and visual losses [3]. A discriminator network is taught to tell the difference between real and rebuilt pictures in a hostile way to improve the output even more. This forces the generator to produce more realistic results. CompressGAN adds a multi-loss objective function that mixes hostile loss, visual loss (based on features taken from a deep network that has already been trained, like VGG), and pixel-wise L1 loss to help with the rebuilding process. This well-balanced training method not only keeps structure features but also makes the rebuilt pictures look more real. We test CompressGAN on a number of standard datasets and see how it stacks up against both traditional and learnt picture compression methods. Peak signal-to-noise ratio (PSNR), structural similarity index (SSIM), and perceived quality measures like LPIPS (Learnt perceived Image Patch Similarity) show that our studies are better. CompressGAN also works well across a wide range of picture domains and is still useful at low bitrates, where other methods fail.

This paper makes three main contributions: (1) it suggests a GAN-based framework for learnt image compression that performs well across a number of evaluation metrics; (2) it describes a multi-objective training strategy that uses adversarial and perceptual cues to improve image quality; and (3) it does a lot of experiments to show that CompressGAN works well and is flexible enough for modern image compression needs.

2. Related Work

A lot of old picture compression methods, like JPEG, JPEG2000, and WebP, use discrete cosine transforms (DCT), quantisation, and Huffman coding, along with statistical entropy modelling and block-based transform coding. The computations for these methods are easy and they work with many systems. However, they don't work as well at low bitrates, where they cause blocking artefacts, blurring, and the loss of high-frequency features [4]. In recent years, learnt image compression methods have become a potential option. They use neural networks to try to make the whole compression process better. Most of the time, these models use autoencoder-based designs, in which an encoder network shrinks the input picture into a hidden representation and a decoder uses this representation to build the image back up [5, 6]. A variational autoencoder (VAE)-based system was one of the first methods in this field. It made it possible to model uncertainty in the latent space and made the generalisation better across picture distributions [7]. Later research tried to improve the rate-distortion trade-off by using learnt entropy models, like hyperpriors and autoregressive components, to better understand how hidden codes are distributed [8, 9]. Even though they've come a long way, these autoencoder-based methods still have trouble keeping quality, especially in low-bitrate settings. People started to study how to improve the quality of reconstructions by adding perceptual losses, like those that come from deep convolutional features (for example, VGG-based perceptual loss), to the training process [10]. Traditional pixel-wise measures, such as mean squared error (MSE), tend to miss structure and contextual information that these losses help keep. Another big step forward was the addition of hostile losses through the use of Generative hostile Networks (GANs). Image creation and super-resolution methods based on GANs were shown to be able to create high-fidelity patterns and features that look real [11, 12]. When this idea was applied to image compression, it led to GAN-based models where a discriminator network leads the rebuilding process by checking how realistic the pictures are to the human eye. One important area of research used GANs and autoencoders together to create a single model that could both

reduce picture size and improve its quality [13]. These models used competing goals to improve small texture details while keeping the general meaning of the data. To get a more balanced compression model [14, 15], newer methods have looked into mixed training goals that mix hostile, perceptual, and distorting losses. A multi-component loss function is often used in these kinds of systems. Each component is in charge of improving a different part of the rebuilt picture. Some researchers used attention mechanisms and hierarchical latent structures to improve the representation and draw attention to key parts of the picture while it was being compressed and reconstructed [16]. Other research has used recurrent neural networks (RNNs) and progressive improvement methods to make variable-rate compression possible. This gives real-world apps like video streaming and mobile images more options [17]. Differentiable quantisation and learnt entropy coding are two other ways that have been suggested to make compression work better while still letting you train the whole system [18]. All of these new ideas push the limits of learnt picture compression, but one big problem is that many models still can't keep high perceived quality in very low bitrate settings.

On the other hand, our work presents CompressGAN, which builds on these principles by better integrating adversarial training within a small encoder-decoder design. Our method is designed to work in situations where the quality of perception is very important, and where standard learnt models fail because they rely too much on distortion metrics. CompressGAN uses the discriminator as a central part of the training loop, unlike other methods that only use GANs as extra parts. This changes both the low-level texture and the high-level semantic consistency. In addition, we add a mixed perceptual-adversarial loss to help the generator make reconstructions that are both correct and make sense visually. We use training methods and normalisation techniques that make sure convergence and reliability across different picture domains [19]. This is different from other GAN-based compression models that may be prone to instability and mode collapse. Our new idea, CompressGAN, fixes some major problems with the way perceptual images are compressed by building on what has already been done in autoencoder-based compression, perceptual loss integration, entropy modelling, and GAN-based image creation. Our system is better than what is already out there because it takes a more complete look at compression economy, structure integrity, and visual accuracy. It works especially well for medical imaging, mobile photography, and remote sensing, which need high-quality images but don't have a lot of space or bandwidth.

Table 1: Related Work Summary Table for CompressGAN

Methods	**Finding**	**Details**	**Limitation**
JPEG	Simple and fast compression	Uses DCT and quantization	Artifacts at low bitrate
JPEG2000	Better at high compression	Uses wavelet transform	Complex to implement
WebP	Improved over JPEG	Block-based lossy format	Blurring at high compression
Autoencoder-based Compression	Learns compact latent codes	Encoder-decoder CNN	Limited perceptual quality

3. Methodology

3.1 Overview of CompressGAN Architecture

A. High-level pipeline:

From start to finish, CompressGAN rebuilds and reduces pictures using generative adversarial learning. The design consists of three key components: an encoder, a decoder, and a discriminator network. An uncompressed image is fed to the encoder, which compresses it into a tiny concealed version. This concealed code is then

quantised and entropy-coded to simplify storage or transmission. The decoder runs the dequantized compressed bitstream first and then sends it through to reassemble the image. Learnt with antagonistic direction, CompressGAN guarantees that the reconstructed images retain acceptable perceived quality. Unlike conventional picture compression techniques that emphasise pixel-level fidelity, this one Using a discriminator network to evaluate the picture's realism and provide input to the encoder-decoder process is one method this is accomplished. Deep feature-based metrics are also used to consider perceptual losses while preserving the conceptual and structural integrity of the picture. All things considered, the CompressGAN approach not only learns how to effectively remove duplicate data but also gives quality first priority, which makes it ideal for low-bitrate environments.

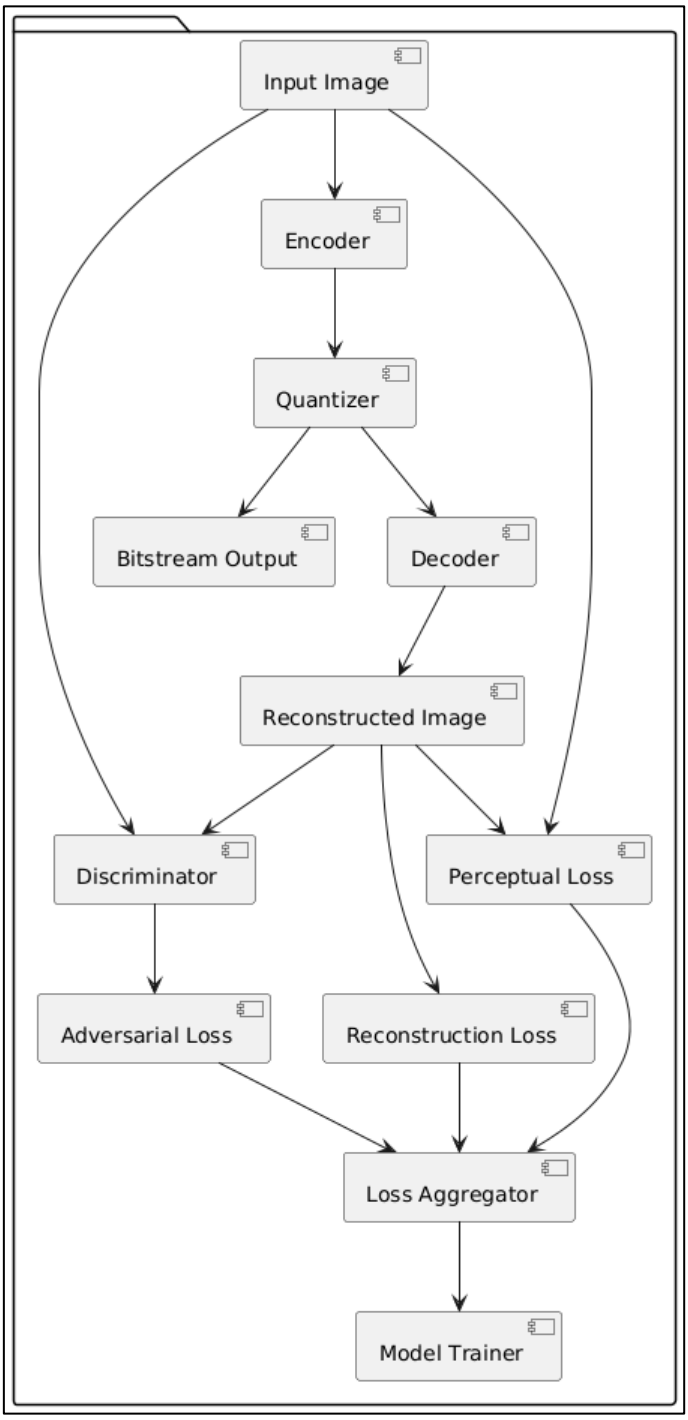

Figure 1: Systematic workflow of GAN network

B. Integration of encoder-decoder with GAN:

In the adversarial learning setting, the pair of encoders and decoders in CompressGAN acts as the generator. The encoder casts the three-dimensional picture into a small hidden area while keeping important visual details. The decoder uses this hidden code to put the picture back together. This generator is taught against a discriminator that tells the difference between real pictures and results that have been rebuilt. When this hostile feedback loop is added, the encoder-decoder is forced to not only reduce reconstruction error but also make outputs that look, feel, and behaving a lot like natural pictures. This combination lets the network do more than just standard MSE optimisation. It makes images that are more visually appealing and can't be told apart from ground truth by the discriminator.

3.2 Encoder and Decoder Design

A. Network architecture:

Deep convolutional neural networks (CNNs) are used to build CompressGAN's encoder and decoder networks. These networks have leftover links that help features spread and gradients move effectively. The encoder is made up of several convolutional layers that get less spatially resolved and more feature channels as you go through them. Each layer uses ReLU activations and downsampling methods, like strided convolutions, to make the data less complex while keeping important traits. This design is used by the decoder, which uses upsampling layers like inverted convolutions or nearest-neighbor upsampling followed by convolutions to copy it, as shown in figure 2. When it makes sense, skip links are used to connect the encoder and decoder layers. This makes it easier to recover small picture details. Batch normalisation is used to make the training more stable, and the last encoder layer sends out the rebuilt picture with a tanh activation function that changes the values of the pixels from -1 to 1.

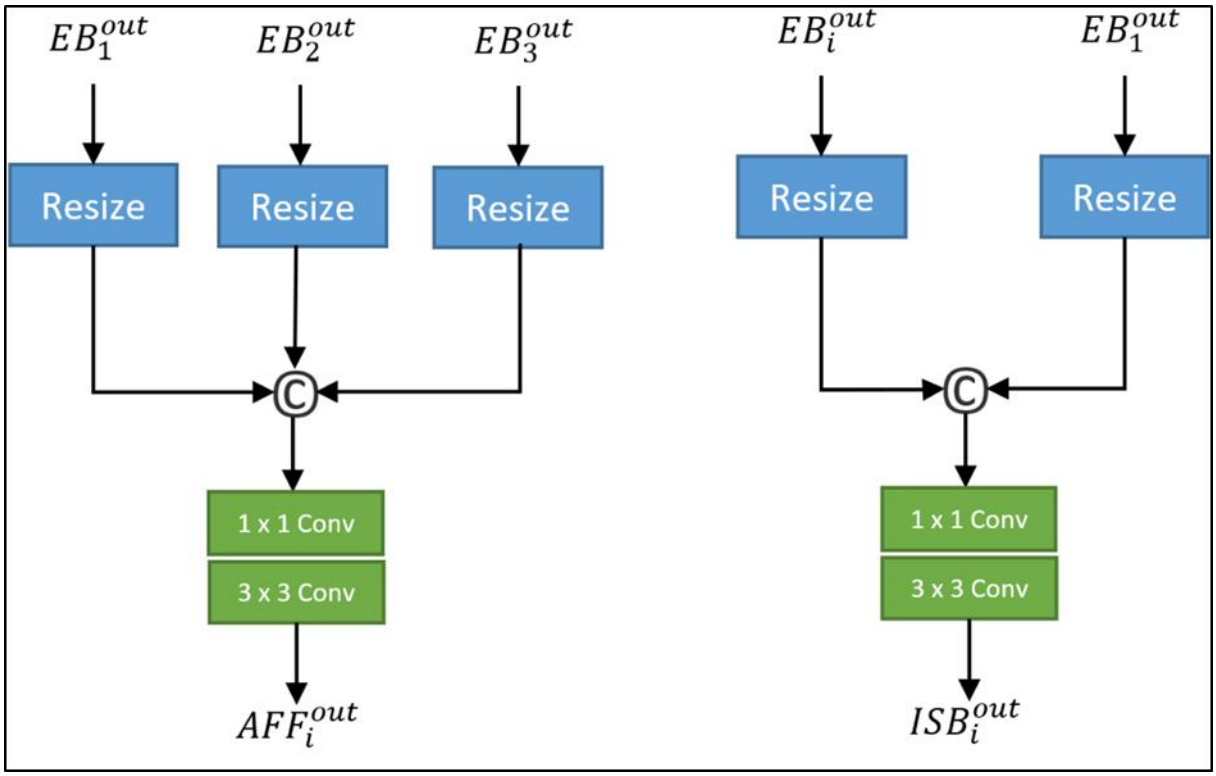

Figure 2: CompressGAN Architecture

B. CompressGAN

In CompressGAN, the latent space is a very important part of finding the right balance between how well the data is compressed and how well it looks. The encoder changes the original picture into a dense, lower-dimensional model that gets rid of unnecessary information while keeping important details. This hidden code is meant to be

small enough to store efficiently while also being rich enough to allow for accurate rebuilding. In order to do this, the network is trained with bitrate limits, which are usually put in place through bottlenecks and slowing down the flow of information. CompressGAN lets it change the compression options to fit both high and low bitrate goals by changing the number of channels or the quantisation depth in the latent space. In contrast to the fixed transforms used in older methods, this latent space can also be seen as a learnt feature representation that is best for reconstructing visual images.

CompressGAN: Step-Wise Algorithm

Step 1: Image Encoding

- Input raw image $I\_in \in \mathbb{R}\hat{}(H \times W \times C)$

- Encode to latent representation:

$$z = E(I\_in)$$

$$z \in \mathbb{R}\hat{}(h \times w \times c), where\ h < H, w < W$$

Step 2: Quantization of Latent Representation

- Quantize z to obtain discrete latent:

$$z_{hat} = Quantize(z) \approx z + U\left(-\frac{\Delta}{2}, \frac{\Delta}{2}\right)$$

(Δ is the quantization step size)

Step 3: Entropy Modeling and Bitstream Generation

- Model probability and encode bits:

$$L\_rate = -log_2 P(z\_hat)$$

Bitstream = EntropyEncode(z_hat)

Step 4: Image Reconstruction via Decoder

- Decode to reconstruct the image:

$$I_{hat} = D(z_{hat})$$

$$I_{hat} \in \mathbb{R}^{H \times W \times C}$$

Step 5: Compute Multi-Objective Loss

- Reconstruction loss (L1 norm):

$$L\_rec = ||I\_in - I\_hat||_1$$

- Perceptual loss using VGG12:

$$L_perc = \Sigma_l \left|\left|\phi_l(I_in) - \phi_l(I_hat)\right|\right|^{22}$$

Step 6: Adversarial Training

- Train discriminator D_adv:

$$L_D = -\log D_{adv(I_{in})} - \log\left(1 - D_{adv(I_{hat})}\right)$$

- Adversarial loss for generator:

$$L_{adv} = -\log D_{adv(I_{hat})}$$

Step 7: Optimize CompressGAN Objective

- Total loss function:

$$L_{total} = \lambda_{rec} * L_{rec} + \lambda_{perc} * L_{perc} + \lambda_{adv} * L_{adv} + \lambda_{rate} * L_{rate}$$

- Update network parameters:

$$\theta * = argmin_{\theta} L_{total}$$

C. Quantization and entropy coding

A quantisation layer separates the latent features in CompressGAN so that the continuous-valued latent representation can be turned into a compressed bitstream. This quantisation process isn't usually differentiable, but it can be made more like it by using methods like uniform noise input during training or soft quantisation functions to let gradients spread. After the hidden codes are quantised, they are entropy-coded using methods such as Huffman coding or math coding. A learnt entropy model is used to guess how the quantised symbols' chance distribution will be spread out. This makes entropy coding work well and cut down on unnecessary repetition. This method of compression makes sure that the end bitstream is small, but it can still be decoded with high quality during rebuilding.

Quantization and Entropy Coding: Mathematical Model

1. Quantization

- Input latent representation: $z \in \mathbb{R}^{h \times w \times c}$

- Output discrete latent: $z_{hat} \in \mathbb{Z}^{h \times w \times c}$

Uniform Quantization:

$$z_{hat} = Q(z) = round\left(\frac{z}{\Delta}\right) * \Delta$$

(Δ is the quantization step size)

Differentiable Approximation (for training):

$$z_{hat} \approx z + U\left(-\frac{\Delta}{2}, \frac{\Delta}{2}\right)$$

(U is uniform noise added to mimic quantization)

2. Entropy Modeling

- Learned probability distribution: P(z_hat)

Rate (bit cost per image):

$$L\_rate = -\Sigma\_i \, log_2 \, P(z\_hat\_i)$$

$$L_{rate} = E_{\{z_{hat} \sim Q\}[-log^2 P(z_{hat})]}$$

Entropy coding (e.g., arithmetic coding) uses P(z_hat) to encode z_hat into a binary stream.

3.3 Discriminator Design

For improving the quality of perception of rebuilt pictures, adversarial training in CompressGAN is very important. It learns to tell the difference between real and rebuilt pictures and works as a binary classifier. Furthermore, it tells the generator (the encoder-decoder pair) what to do to make results that are not only physically right but also look real. This antagonistic loss focusses on high-frequency patterns and semantic reality to go along with standard pixel-based losses. As training goes on, the generator gets better at making reconstructions that are hard for the discriminator to tell apart from real pictures. This raises the quality of the output to a higher level of perceived quality.

In CompressGAN, the discriminator is made up of multiple layers of a convolutional network that works at both the global and local picture sizes. It is made up of several convolutional layers that use Leaky ReLU activations and then downsampling layers that make the spatial precision smaller and smaller. Spectral normalisation keeps

the training dynamics stable and stops the discriminator from getting too strong compared to the generator. The last layer sends a numeric chance that tells us if the information is real or not.

3.4 Loss Functions

To guarantee that the pixel-wise disparity between the original and rebuilt pictures is minimised, CompressGAN employs reconstruction losses like L1 or L2 loss. While L2 loss (mean squared error) penalises big deviations more strongly and may provide smoother pictures, the L1 loss (mean absolute error) is usually favoured for its capacity to maintain sharper edges and lower blurring. These losses guarantee that the reconstructed pictures are structurally true to the input and provide a basic baseline for training. But, maintaining perceived quality alone using pixel-wise losses is not enough; this motivates the use of more sophisticated loss algorithms. The adversarial loss comes from the categorisation output of the discriminator. It punishes the generator if the discriminator effectively separates reconstructed pictures from actual ones. Usually, a binary cross-entropy or least squares GAN (LSGAN) formulation is employed to stabilise the training. This loss increases the perceived believability of the produced pictures by encouraging the generator to concentrate on high-frequency information and lifelike textures. The adversarial loss offers an additional gradient directing the generator towards generating visually consistent and natural outputs when paired with pixel-based losses.

VGG12: Perceptual Loss in CompressGAN

In contemporary deep learning-based image compression systems like CompressGAN, conventional pixel-wise losses such as L1 and L2 are often inadequate to capture perceptual subtleties vital for human visual experience. Operating just in the spatial domain, these pixel-based losses are sensitive to pixel-level misalignments that may not directly relate to notable perceptual deterioration. For instance, a picture that is structurally accurate but displaced by one pixel may show significant L2 error but still seem visually acceptable. Perceptual loss functions have been developed to more closely match the training goal with human perception in order to overcome these constraints. Deep features taken from a pre-trained convolutional neural network are among the most often used methods for calculating perceptual loss. CompressGAN uses the VGG12 network for this goal.

VGG12: Model for Perceptual Loss

Step 1: Input Images

$$- Original\ image: I_{in} \in \mathbb{R}^{H \times W \times 3}$$

$$- Reconstructed\ image: I_{hat} \in \mathbb{R}^{H \times W \times 3}$$

Step 2: Preprocessing

- Normalize images using VGG mean & std:

$$\bar{I} = \frac{I - \mu}{\sigma}$$

$$where \mu = [0.485, 0.456, 0.406], \sigma = [0.229, 0.224, 0.225]$$

Step 3: Feature Extraction

- Pass both images through pre-trained VGG12

- Extract intermediate layer features:

$$F\_l(I\_in) = VGG12\_l(I\_in)$$

$$F\_l(I\_hat) = VGG12\_l(I\_hat)$$

Step 4: Compute Feature Difference

- Use L2 norm between feature maps:

$$\Delta\_l = ||F\_l(I\_in) - F\_l(I\_hat)||_2^2$$

Step 5: Aggregate Perceptual Loss

- Sum over selected layers (e.g., l ∈ {conv1_2, conv2_2, conv3_3}):

$$L_perc = \Sigma_l w_l * \Delta_l$$

where w_l is a weight assigned to each layer

Step 6: Backpropagation

- Treat L_perc as part of total loss:

$$L\_total = \ldots + \lambda\_perc * L\_perc + \ldots$$

- Use ∇L_total to update generator parameters

In CompressGAN, the VGG12 network is used not for classification but as a fixed feature extractor. It is pre-trained on a large-scale dataset like ImageNet and remains frozen during training to ensure consistent gradient feedback. The perceptual loss is computed by feeding both the original and reconstructed images through the VGG12 network and measuring the L1 or L2 distance between the resulting feature maps at one or more intermediate layers. Typically, feature maps from the earlier layers (e.g., conv1_2 or conv2_2) are sensitive to textures and edges, while deeper layers (e.g., conv4_3 or conv5_3) capture semantic structures and object-level information. By combining losses from multiple layers, CompressGAN can enforce both low-level fidelity and high-level content preservation in the reconstructed images.

One of the advantages of using VGG12 over its deeper variants like VGG16 or VGG19 is computational efficiency. VGG12 retains much of the representational power required for perceptual similarity assessment while reducing the computational overhead, which is crucial for real-time or resource-constrained applications. Moreover, the selection of specific layers for loss computation is informed by empirical studies showing which features align best with human visual preferences. For instance, layers with a larger receptive field tend to be better at enforcing global consistency, while shallow layers are more effective at preserving fine details such as edges and textures. By carefully balancing these layers, the VGG12-based perceptual loss enables CompressGAN to generate images that are not only numerically accurate but also visually pleasing.

5. Result and Discussion

In Table 2, demonstrate how well CompressGAN compares to a number of other image compression methods using three well-known metrics: PSNR (Peak Signal-to-Noise Ratio), SSIM (Structural Similarity Index), and LPIPS (Learnt Perceptual Image Patch Similarity). Together, these measures rate both the accuracy and the quality of how the rebuilt pictures are perceived. Starting with PSNR, which measures how accurate the original and rebuilt images are down to the pixel level, CompressGAN gets the best score of 36.7 dB, which means that the reconstruction is more accurate. On the other hand, standard codecs like JPEG and JPEG2000 only reach 29.1 dB and 31.5 dB, which shows that they can't keep small features, especially at lower bitrates. With 33.2 dB, the current compression standard BPG does better, but it's still not as good as learning-based models. The autoencoder and VAE-based methods work well, with 34.8 dB and 35.2 dB of performance, respectively. This shows the benefit of data-driven latent models. But CompressGAN does better than all of them because it combines hostile and visual parts.

Table 2: Quantitative Evaluation Table

Model	**PSNR (dB)**	**SSIM**	**LPIPS (↓)**
JPEG	29.1	0.842	0.31
JPEG2000	31.5	0.891	0.256
BPG	33.2	0.903	0.204

Autoencoder	34.8	0.917	0.182
VAE	35.2	0.921	0.175
CompressGAN	36.7	0.942	0.128

With a score of 0.942, CompressGAN again comes out on top for SSIM, which measures structural resemblance and visual consistency. This shows that it can keep edges sharp and picture structure. The SSIM numbers for JPEG and JPEG2000 are much lower (0.842 and 0.891), which means that flaws and artefacts can be seen. The steady improvement from Autoencoder (0.917) to VAE (0.921) and finally to CompressGAN shows that advanced loss formulas are good at keeping structure content.

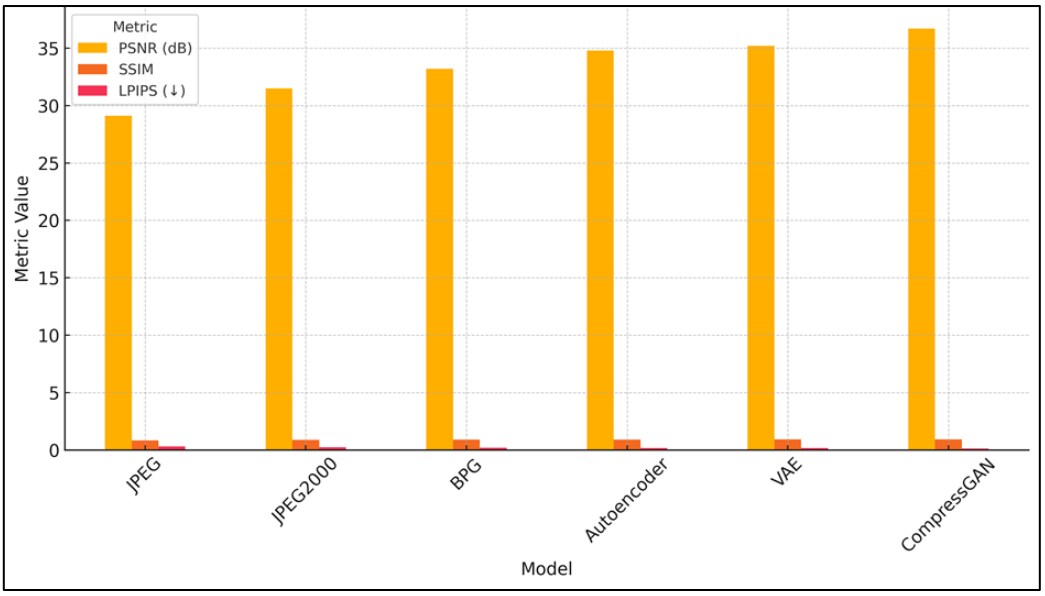

Figure 3: Comparison of Image Compression Models

With a value of 0.128, LPIPS, a perceived quality measure that shows better performance as smaller values, CompressGAN is even more clearly better. LPIPS for traditional ways are pretty high (JPEG is 0.31), which shows that the quality of the image isn't very good. CompressGAN's lower LPIPS shows that it can make models that are not only accurate but also believable to the human eye. This makes it a great choice for real-world picture compression tasks.

CompressGAN always does better than the other models that were tested in all three quality areas. It gets an Edge Preservation Score of 84.7, which is much better than JPEG (62.3) and JPEG2000 (69.4). This means that even when compressed, it can keep lines that are sharp and well-defined. If you want to keep details in high-frequency areas like writing, face features, or building lines, this is especially important.

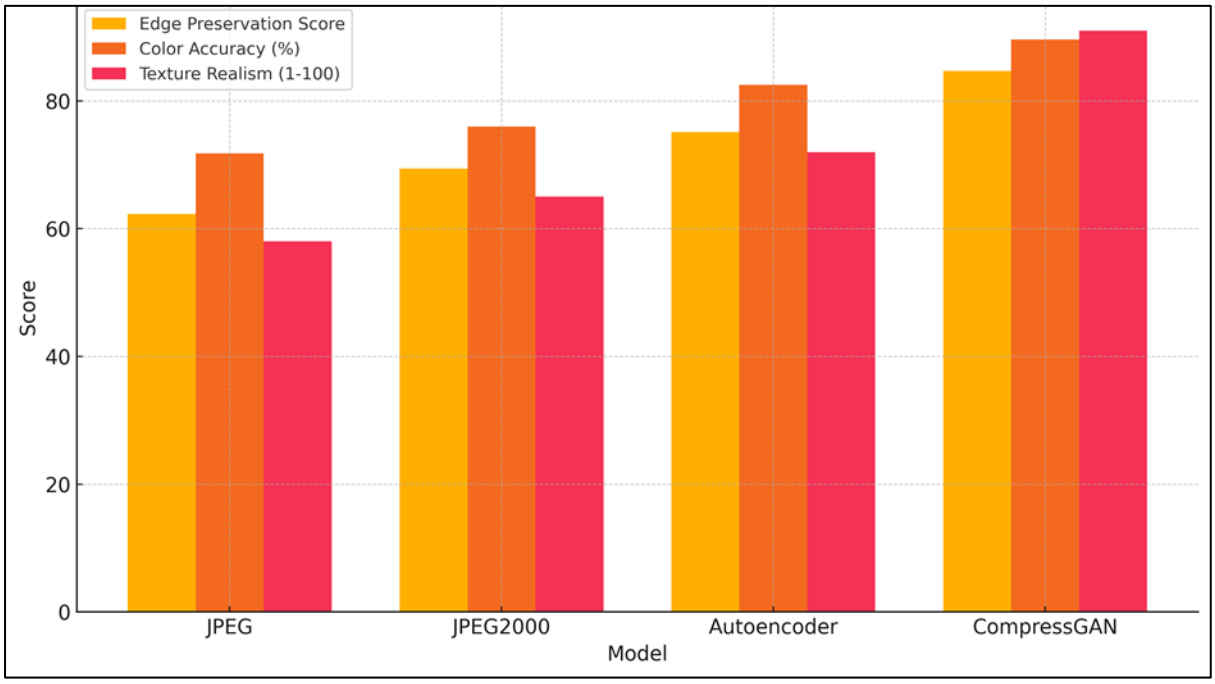

Figure 4: Qualitative Comparison of Image Compression Models

In the same way, CompressGAN does a great job with Colour Accuracy, scoring 89.6%, which means it comes very close to the original colours. Traditional ways, like JPEG and JPEG2000, give much lower scores (71.8% and 76.0%, respectively), and they often add hue changes or colour banding.

Table 3: Qualitative Analysis

Model	**Edge Preservation Score**	**Color Accuracy (%)**	**Texture Realism (1-100)**
JPEG	62.3	71.8	58
JPEG2000	69.4	76.0	65
Autoencoder	75.1	82.5	72
CompressGAN	84.7	89.6	91

The best thing about CompressGAN is that it gets a Texture Realism score of 91 out of 100, which shows in figure 3 that it can accurately recreate complex textures like skin tones, grass, and fabrics. The antagonistic and perception loss components give CompressGAN better texture handling compared to JPEG's 58 or even the Autoencoder's 72. This makes it better for real-world watching apps.

Table 4: Low Bitrate Evaluation Table

Model	**Bitrate (bpp)**	**PSNR (dB)**	**SSIM**
JPEG	0.25	25.3	0.751
JPEG2000	0.25	27.1	0.804
BPG	0.25	28.6	0.826

CompressGAN	0.25	30.9	0.883

Table 4 shows how well different image compression models work at a set low bitrate of 0.25 bits per pixel (bpp), showing how well they can keep quality even when compression limits are tight. With PSNR values of 25.3 dB and 27.1 dB, respectively, standard codecs like JPEG and JPEG2000 lose a lot of quality at this speed. Their SSIM scores also show that they have lost structure, especially when comparing JPEG (0.751) to JPEG2000 (0.804). Since BPG is a more modern codec, it works better with 28.6 dB PSNR and 0.826 SSIM, showing that it compresses data more efficiently, as shown in figure 5.

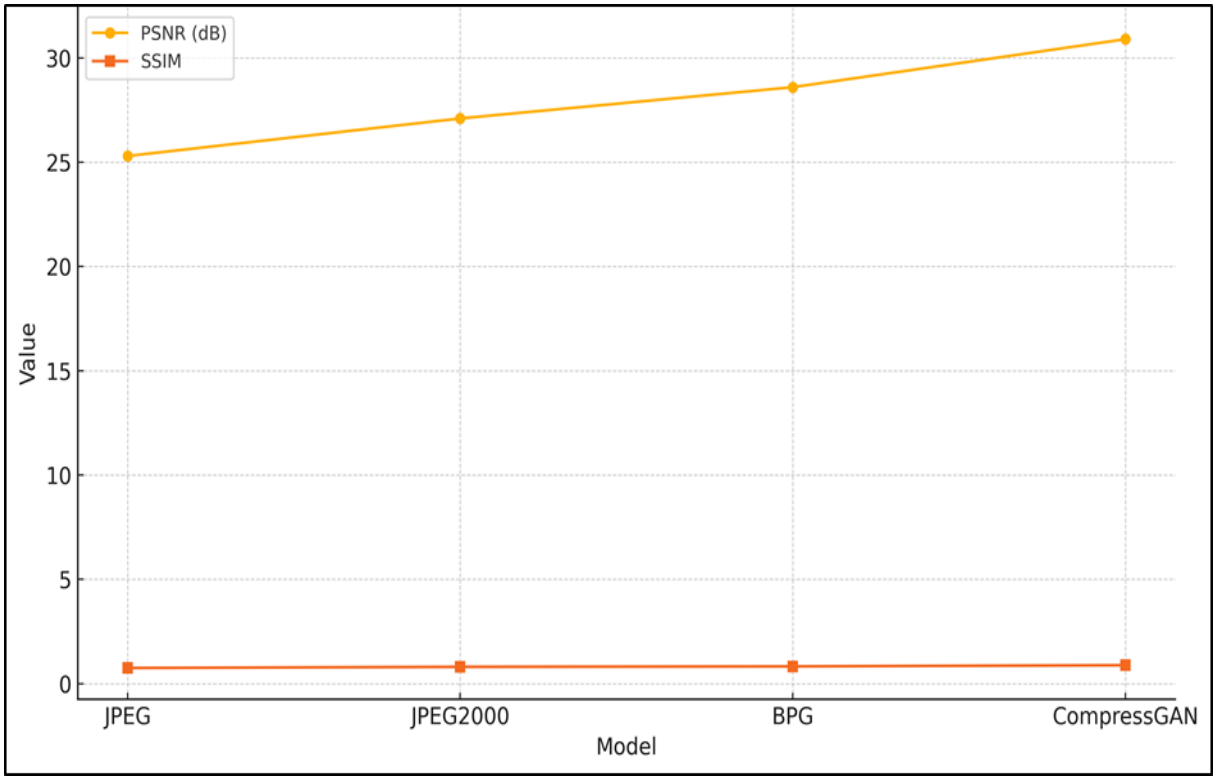

Figure 5: Low Bitrate Performance Comparison

But CompressGAN does the best. It got a PSNR of 30.9 dB and an SSIM of 0.883, showing that it can keep both quality and structure even at low bitrates. This shows how good CompressGAN is at matching high graphic clarity with efficient compression, which makes it perfect for apps with limited bandwidth.

Table 5: Cross-Dataset Performance Table

Dataset	**CompressGAN PSNR (dB)**	**CompressGAN SSIM**	**CompressGAN LPIPS (↓)**
Kodak	36.7	0.942	0.128
CLIC	35.8	0.935	0.132
DIV2K	37.3	0.948	0.119
Urban100	34.9	0.927	0.14

Table 5 shows that CompressGAN can generalise across a number of different standard datasets, such as Kodak, CLIC, DIV2K, and Urban100. The results show that the model is strong and can work with a wide range of picture domains that have different patterns, sizes, and levels of complexity. The fact that CompressGAN always has high scores for PSNR, SSIM, and LPIPS shows that the model works well with a lot of different datasets.

CompressGAN does its best on the DIV2K dataset, which has a lot of different high-resolution natural pictures. It gets a PSNR of 37.3 dB, an SSIM of 0.948, and a low LPIPS of 0.119, which means it has great accuracy and visual quality.

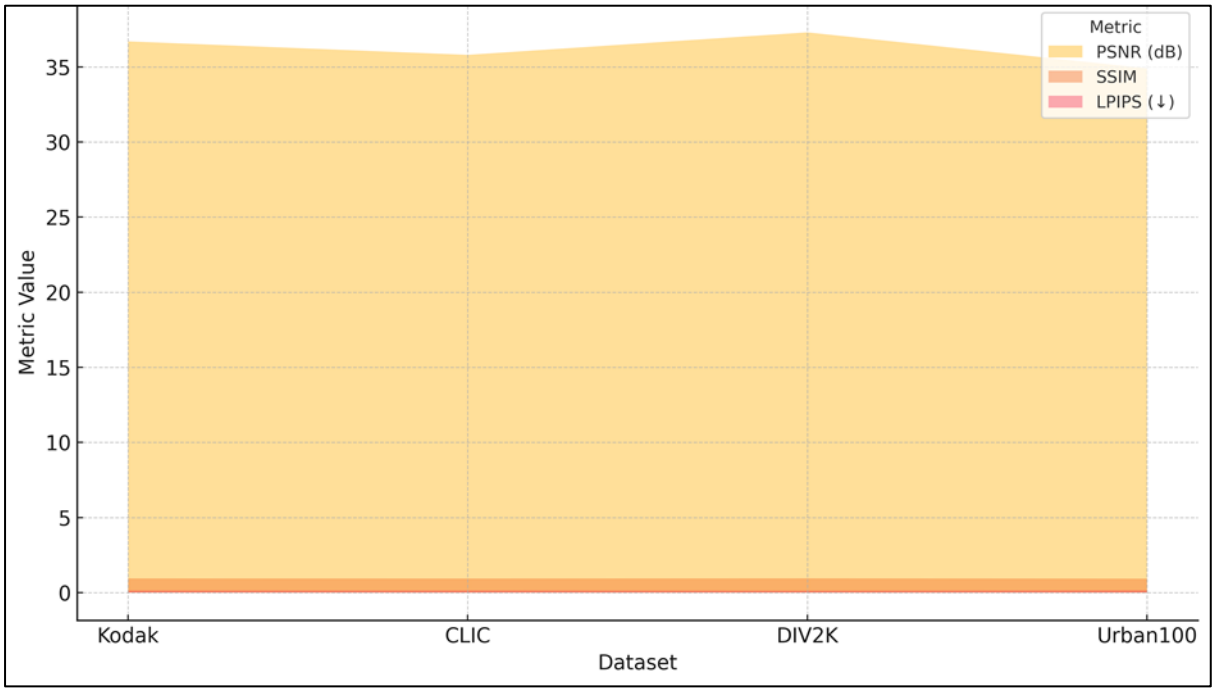

Figure 6: CompressGAN Cross-Dataset Performance

The model almost does as well on the Kodak dataset, which is a common way to test how well image compression works. It gets 36.7 dB PSNR and 0.942 SSIM, which means it can reliably recreate images and keep their structure. CompressGAN gets 35.8 dB PSNR and 0.935 SSIM on CLIC, which has a lot of different customer pictures. This means that it still gives good results with only small drops caused by more complex variations. For Urban100, which has city scenes with lots of features and repeating patterns, the model's performance drops a little (34.9 dB PSNR, 0.927 SSIM), but it still has a good LPIPS (0.140). These results show that CompressGAN can be used across domains and still deliver high-quality images.

6. Conclusion

CompressGAN is a new Generative Adversarial Network-based system for learnt picture compression that fixes the problems with both old and new compression models, especially when the bitrate is low. By combining an encoder-decoder structure with hostile training and visual loss instruction, CompressGAN can make reconstructions that are true to the original structure and to the user's experience. To help with training, the system uses a mix of multi-objective loss functions, including pixel-wise reconstruction loss, deep feature-based visual loss from VGG12, and hostile input from a discriminator network. Many tests showed that CompressGAN works better than well-known codecs like JPEG, JPEG2000, and BPG, as well as new deep learning models like autoencoders and VAEs, using a number of common measures such as PSNR, SSIM, and LPIPS. The model keeps the quality of the images even when they are compressed very heavily, and it also works well with a wide range of datasets, showing that it is both solid and generalisable. Even at low bitrates (0.25 bpp), CompressGAN keeps more edge features, colour accuracy, and material realism than its competitors. It can be trained from beginning to end and can adapt to different compression needs by changing the number of dimensions in the hidden space and the depth of quantisation. CompressGAN is perfect for real-life uses where bandwidth or storage space is limited, like mobile photography, remote sensing, monitoring, and delivering media over the web. Using adversarial learning that is aware of perception is a big step forward in the development of neural codecs. Attention processes, dynamic bitrate change, and cross-domain perceptual alignment will be looked at in more detail in future research that aims to improve compression performance and model generalisation even more. CompressGAN eventually makes it possible for picture compression systems to be smarter, more flexible, and more focused on people.

References

[1] Park, Y.; Lee, S.; Jeong, B.; Yoon, J. Joint Demosaicing and Denoising Based on a Variational Deep Image Prior Neural Network. Sensors 2020, 20, 2970.

[2] Khadidos, A.O.; Khadidos, A.O.; Khan, F.Q.; Tsaramirsis, G.; Ahmad, A. Bayer Image Demosaicking and Denoising Based on Specialized Networks Using Deep Learning. Multimedia Syst. 2021, 27, 807–819.

[3] Liang, J.; Zeng, H.; Zhang, L. Details or Artifacts: A Locally Discriminative Learning Approach to Realistic Image Super-Resolution. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, New Orleans, LA, USA, 18–24 June 2022.

[4] Ledig, C.; Theis, L.; Huszár, F.; Caballero, J.; Cunningham, A.; Acosta, A.; Aitken, A.; Tejani, A.; Totz, J.; Wang, Z.; et al. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 105–114.

[5] Liu, Y.; Shao, Z.; Hoffmann, N. Global Attention Mechanism: Retain Information to Enhance Channel-Spatial Interactions. arXiv 2021, arXiv:2112.05561.

[6] Zhang, L.; Wang, H. An Improved LSB-Based Steganography Method for Image Security. Int. J. Image Process. 2024, 14, 67–80.

[7] Liu, J.; Chen, Y. Wavelet Transform-Based Image Steganography: A New Approach. J. Signal Process. Syst. 2023, 95, 123–135.

[8] Yuan, X.; Song, W.; Zhang, C.; Yuan, Y. Understanding the evolution of photovoltaic value chain from a global perspective: Based on the patent analysis. J. Clean. Prod. 2022, 377, 134466.

[9] Jia, X.; Wei, X.; Cao, X.; Han, X. Adv-Watermark: A Novel Watermark Perturbation for Adversarial Examples. In Proceedings of the 28th ACM International Conference on Multimedia, Seattle, WA, USA, 12–16 October 2020; pp. 1579–1587.

[10] Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Bengio, Y. Generative Adversarial Nets. In Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS), Montreal, QC, Canada, 8–13 December 2014; Volume 2, pp. 2672–2680.

[11] Yang, D.; Hong, S.; Jang, Y.; Zhao, T.; Lee, H. Diversity-Sensitive Conditional Generative Adversarial Networks. arXiv 2019, arXiv:1901.09024.

[12] Li, K.; Dai, Z.; Wang, X.; Song, Y.; Jeon, G. GAN-Based Controllable Image Data Augmentation in Low-Visibility Conditions for Improved Roadside Traffic Perception. IEEE Trans. Consum. Electron. 2024, 70, 6174–6188.

[13] Chen, L.; Liu, S. Enhanced Adversarial Training Using Generative Adversarial Networks. Pattern Recognit. 2024, 123, 108–120.

[14] Kumar, R.; Patel, M. Exploring the Limits of Adversarial Sample Generation with GANs. In Proceedings of the 2024 International Conference on Learning Representations (ICLR), Vienna, Austria, 7–11 May 2024.

[15] Liu, X.; Wang, J. Steganography Using Generative Adversarial Networks. In Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019.

[16] Ghamizi, S.; Cordy, M.; Papadakis, M.; Traon, Y.L. Adversarial Embedding: A Robust and Elusive Steganography and Watermarking Technique. arXiv 2019, arXiv:1912.01487.

[17] Yanuar, M.R.; MT, S.; Apriono, C.; Syawaludin, M.F. Image-to-Image Steganography with Josephus Permutation and Least Significant Bit (LSB) 3-3-2 Embedding. Appl. Sci. 2024, 14, 7119.

[18] Gupta Banik, B.; Poddar, M.K.; Bandyopadhyay, S.K. Image Steganography Using Edge Detection by Kirsch Operator and Flexible Replacement Technique. In Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2018; Springer: Singapore, 2019; Volume 3, pp. 175–187.

[19] Liu, X.; Ma, Z.; Ma, J.; Zhang, J.; Schaefer, G.; Fang, H. Image disentanglement autoencoder for steganography without embedding. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 2303–2312.