Removing Unwanted Noise from Real Scene Images using GANs

DOI: 10.38124/ijsrmt.v3i10.60

Manash Bandhu Barik[1], Sidharth[2]

[1] Department of AIT-CSE Chandigarh University Mohali, India
[2] Department of AIT-CSE Chandigarh University Mohali, India

Abstract

Various forms of noise, including as sensor noise, compression artefacts, and ambient disturbances, are frequently present in real-world photographs. These noises can severely reduce the quality of the images and have an effect on future computer vision tasks. In this study, we offer a unique method that uses Generative Adversarial Networks (GANs) to remove undesired sounds from actual scene photos. Since they can produce realistic pictures and understand intricate data distributions, GANs have become a potent tool in the image production and modification domain. Our approach uses GAN model which is made up of a discriminator and a generator network, where the discriminator's job is to discern between actual and created pictures, while the generator's is to produce clean images from noisy inputs. The generator efficiently learns to eliminate noise from input pictures while maintaining important features and structures using an adversarial training procedure. We assess the suggested method using industry-standard benchmark datasets and show encouraging outcomes in terms of picture restoration and noise reduction. By improving picture denoising algorithms, this research advances potential uses in surveillance analysis, object recognition, and semantic segmentation.

Keywords: *Image Denoising, Generative Adversarial Networks (GANs), Noise Reduction, Image Quality Enhancement, Noisy Image Processing.*

I. INTRODUCTION

Image denoising has become an important task in many computer vision applications in recent years. Noise corruption in digital photographs can result from a variety of sources, including camera sensors, lighting conditions, transmission faults, and compression artifacts. Biological imaging also has difficulties with low light and short exposure times, which deteriorates image quality even further [1]. As a result, image restoration has become essential in a variety of domains, such as underwater.

CNN is a deep learning, have revolutionized image processing applications in recent years. In many image-related applications, such as image identification, object detection, and image denoising, CNNs perform remarkably well. Furthermore, the ability of GANs to produce realistic images has won them notoriety [2].

Motivated by recent developments in deep learning, we suggest a new GAN-based model for picture denoising. Our model learns to counteract picture noise by utilizing the generator and discriminator architecture of GANs. Specifically, our proposed generator employs an encoder-decoder architecture with shortcut connections akin to Res-Net in order to preserve image texture features. To increase training stability, the loss function incorporates the Wasserstein distance [3].

A. Types of Noise in the Image

Noise is the unwanted signals that distort the original signals, here in images, this refers to unwanted variations in pixel values of the images that degrade the quality and hinder accurate interpretation and analysis.

- *Gaussian Noise:*

Gaussian noise is characterized by random variations in pixel intensity levels, following a Gaussian distribution. It often occurs due to sensor limitations, electronic interference, or transmission errors, manifesting as a smooth, continuous distribution of noise across the image.

- *Salt and Pepper Noise:*

This consists of randomly occurring bright and dark pixels scattered throughout the image. It typically arises from sensor malfunctions, transmission errors, or data corruption, leading to isolated high-intensity (salt) and low-intensity (pepper) pixels [4].

- *Shot Noise:*

Shot noise is inherent in photon-based imaging systems and occurs due to the random arrival of photons at the image sensor. It introduces variability in pixel values,

How to Cite: Barik, M. B., & Sidharth. (2024). Removing Unwanted Noise from Real Scene Images using GANs. *International Journal of Scientific Research and Modern Technology*, 3(10). https://doi.org/10.38124/ijsrmt.v3i10.60

particularly in low-light conditions, and appears as grainy or speckled patterns in the image.

➢ *Speckle Noise:*

Speckle noise is prevalent in coherent imaging systems such as ultrasound and synthetic aperture radar (SAR) and arises from the interference of coherent waves. It appears as granular patterns that distort image features and reduce clarity [5].

➢ *Quantization Noise:*

Quantization noise occurs during the digitization process when continuous analog signals are discretized into digital values. It introduces errors due to the finite precision of digital representations, resulting in loss of detail and fidelity in the image.

➢ *Random Noise:*

Random noise encompasses various sources of unpredictable fluctuations in pixel values, including electrical noise, thermal noise, and environmental factors. It can manifest in different forms, such as high-frequency spikes or low-frequency variations, depending on the underlying causes.

II. RELATED WORK

Recent research has explored the use of Generative Adversarial Networks (GANs) for image denoising, demonstrating their effectiveness in preserving texture and detail while removing noise. Various GAN architectures have been proposed, including a new generator network with a novel loss function. ZhiPing, Qu, et al [6] proposed GAN based network with residual dense blocks. This approach has shown superior performance compared to traditional denoising methods. Alsaiari, Abeer, et al [7] have also investigated the use of Wasserstein distance, perceptual loss, and reconstruction loss in the objective function to improve denoising results. Zhong, Yue, et al [8] performed a study in which the multi-level convolution and residual blocks have been incorporated into GAN architectures to enhance feature extraction and learning. Chen, Songkui, et al [9] designed an application for image denoising using GAN. This application has become a significant area of research, with potential for further advancements in the field.

A. Image Denoising

Chen, Jingwen, et al [10] developed an image denoising method. This aims to remove undesirable noise while keeping significant information and features of the image. There are two types of traditional image denoising techniques: spatial and frequency-based.

➢ *Spatial Denoising Methods*

Li, Yuqin, et al [11] proposed a spatial denoising method. This spatial denoising techniques work directly with the image's pixel values. To reduce noise and maintain visual details, these techniques usually entail local averaging or filtering procedures. Typical methods for spatial denoising include of:

- *Median Filtering:*

This non-linear filtering technique effectively eliminates salt-and-pepper noise by substituting the median value of each pixel's vicinity for each one.

- *Gaussian Filtering:*

This technique reduces Gaussian noise by applying a weighted average using a Gaussian kernel to the vicinity of each pixel.

- *Wiener Filtering:*

Thus filter is used to remove the additive noise and it applies the frequencies depending on the weighting transform the image. This process takes place by calculating the noise power spectrum.

➢ *Frequency Denoising Methods:*

Sheng, Zehua, et al [12] developed a denoising method using frequency-domain deep guided method. This method uses filtering based on the frequency features that separate the signal from the noise that are in the images. After applying the Fourier transformations to images, frequency denoising methods will operate in the frequency domain. Some popular methods of frequency denoising are given below:

- *Wavelet Denoising:*

This technique applies thresholding to reduce noise while preserving image details after dividing the images into numerous frequency subbands.

- *Bandpass Filtering:*

This method utilizes the threshold to reduce noise when preserve picture information after dividing the image into many frequency sub bands.

B. CNN- Based Image Processing

Zhang, Kai et al [13] propose CNN-based techniques which have shown to be extraordinarily successful at eliminating noise from images while preserving crucial aspects of the image. These type networks have revolutionized the field of image processing and denoising over the years because they construct hierarchical representations directly from raw pixel input. In these type of networks some convolutional layers, activation functions, pooling layers, and flatten layer are used to build the systems. This network enables to translate the noisy input image to clean image by minimizing suitable loss function like mean squared error or perceptual loss [14]. Training processes involve various steps and it uses large datasets which contain pairs of clean and noisy images. Some other methods like batch normalization and dropout are used to minimize overfitting and enhance generalization of the images [15].

Optimization techniques like Adam and Stochastic Gradient Descent (SGD) are used to ensure the model convergence, divergence and prevent [16]. When assessing CNN-based image denoising models, evaluation metrics such as mean absolute error, peak signal-to-noise ratio and structural similarity index are taken to measure the quality. Advances in CNN-based image processing include

adversarial training methodologies, attention mechanisms, and deep residual networks. Future work will probably concentrate on novel architectures, domain-specific priors, and issues with managing heterogeneous noise and data distributions [17].

III. METHODOLOGY

A. Generative Adversarial Networks

In contemporary studies of deep learning, GANs have come up as a mostly novel avenue, especially in the domain of image generation, manipulation and denoising. The GANs form a minimax game which consists of two neural networks referred to as the discriminator and the generator. The goal of the generator is making synthetic samples that are indistinguishable to the naked positive from the standard original image while on the other hand, the aim of the discriminator is to differentiate the real image from the synthesized fakes.

B. DC Generative Adversarial Networks

A DCGAN is just a simple extension of the Generative Adversarial Networks covered, with the exception of the explicit addition of convolutional and convolutional-transpose layers in the discriminator and generator, respectively. This model can be represented by unsupervised learning with DCGAN. The discriminator is composed of multiple parameters, such as batch norm layers, stride convolution layers, and leaky ReLU activations. The input is a 3x64x64 input picture, and the output is a scalar probability indicating that the input is from the real data distribution. The generator consists of convolutional-transpose layers, batch normalization layers, and ReLU activations. The input is a latent vector, z that is drawn from a traditional normal distribution; the output is a 3x64x64 RGB picture.

C. Residual Learning

Residual learning has emerged as a key deep learning technique aimed at alleviating problems associated with extremely deep neural networks. In an effort to enhance model performance and extract intricate picture characteristics for tasks like image denoising, researchers frequently encounter difficulties with deeper networks, such as information loss during feature extraction. So created the residual structure, which effectively reduces information loss by constructing paths between input and output inside each block through the integration of skip connections. Fig. 1 shows the generator network architecture.

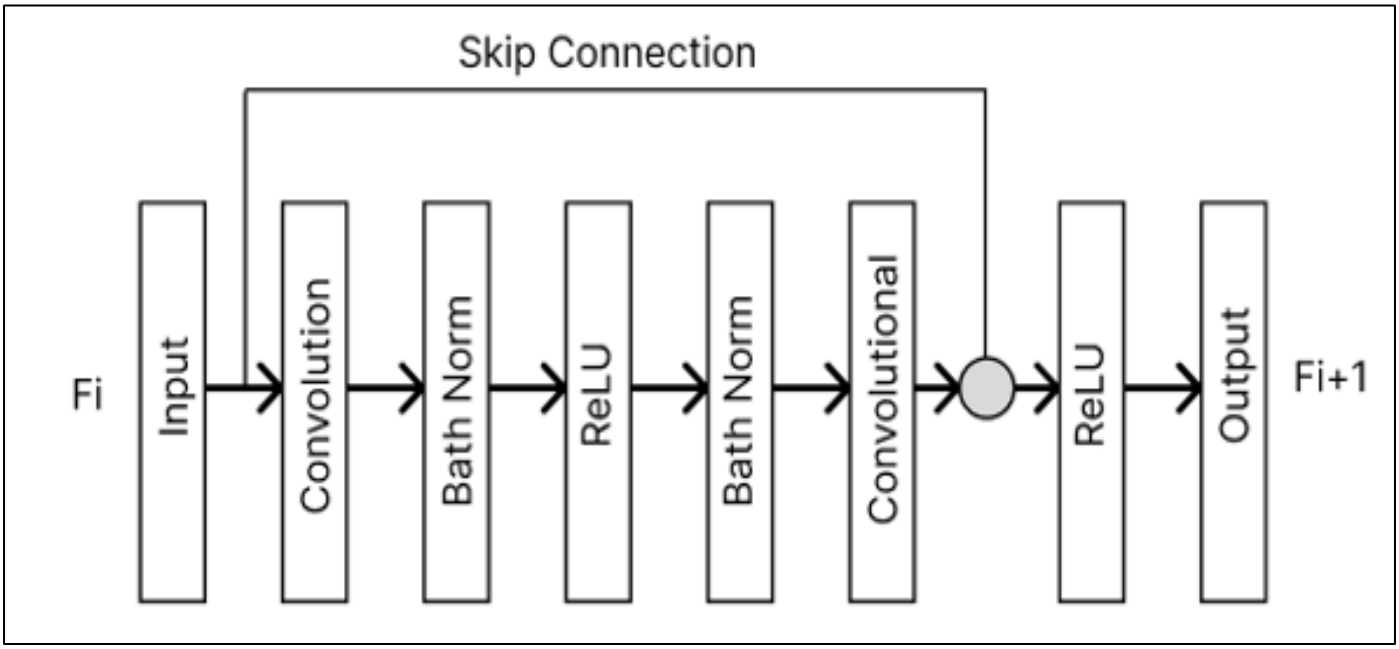

Fig 1 Generator Network Architecture

The generator is denoted by the letter G. A latent space vector, Vz, is transformed by G into data-space, which is made up of picture data. To perform this conversion, an RGB image with the same dimensions as the training images—3x64x64—must be created. This is accomplished by employing a sequence of stride two-dimensional convolutional transpose layers, each of which is succeeded by a two-dimensional batch normalization layer and an activation function represented by rectified linear units (ReLU). The generator's output is additionally subjected to a hyperbolic tangent (tanh) function to make sure it is within the input data range of [−1, 1].

Interestingly, batch of normalization—a crucial technique described in the DCGAN study to promote gradient flow during training—is done following the convolutional transpose layers. The input section's parameters—namely, ngf, nz, and c—have a significant impact on the code's generator design. In this instance, ngf sets the size of the feature maps that propagate through the generator, nc, which stands for the number of channels in the output image and is typically set to three for RGB images, indicates the number of channels in the image, and dz indicates the length of the input of the vector that are given by d. An architectural diagram of the generator is included in the DCGAN paper. Fig. 2 shows the DC-GAN generator network architecture.

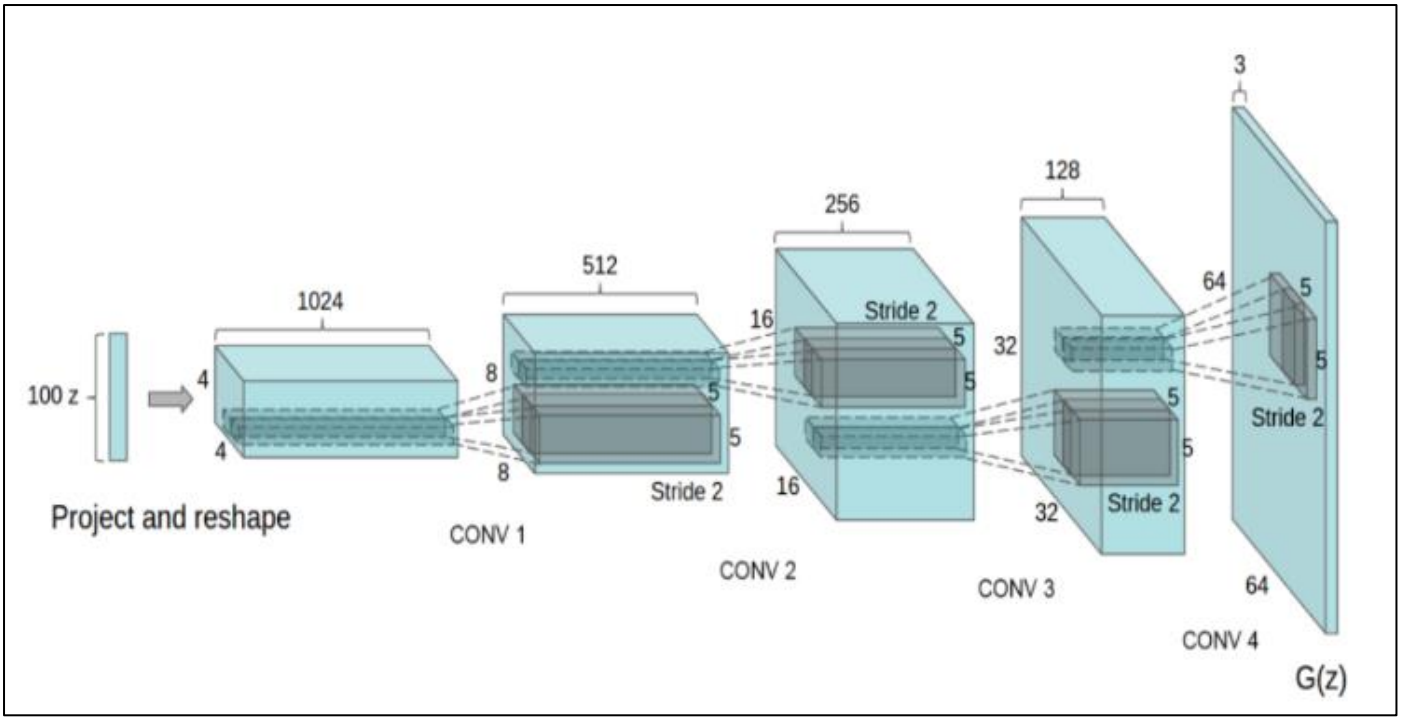

Fig 2 DC-GAN Generator network Architecture

D. Discriminator Network Architecture

As a binary classification network, the discriminator, represented by (D), is tasked with identifying the probability that an input image is authentic or fraudulent. After going through a series of training process by Conv2d, BatchNorm2d, and LeakyReLU layers on a 3x64x64 input image, it outputs a scalar probability using function such as Sigmoid activation function. The selection of the BatchNorm, LeakyReLU, and strided convolution layers is important, even if this architecture can be expanded with further layers as needed. Since strided convolution enables the network to train its own pooling function, the DCGAN article recommends it over pooling for downsampling. Furthermore, the use of LeakyReLU activation functions and batch normalization fosters a healthy gradient flow is necessary for the recoganization, which is essential for the learning process of the generator (G) and discriminator (D).

E. Loss Functions and Optimizers

After the discriminator (D) and generator (G) are set, the learning process is controlled by choosing loss functions and optimizers. To compute the binary cross entropy between the target and predicted labels, we use the PyTorch library's defined Binary Cross Entropy loss (BCELoss) function. The two halves of the objective function is represented by (log(D(x))) and (log(1 - D(G(z))), are combined in this function. We may provide flexibility in the training process by selecting which part of the equation to calculate by changing the target labels (y).

$$\ell(x, y) = L = \{l_1, \ldots, l_N\}^\top, \quad l_n = -\left[y_n \cdot \log x_n + (1 - y_n) \cdot \log(1 - x_n)\right]$$

In order to maintain consistency with the original GAN paper, we refer to the true label as 1 and the false label as 0. Two different Adam optimizers are used for (D) and (G), as suggested in this DCGAN research paper, with settings like learning rate of 0.0002 and Beta1 set to 0.5. Additionally, we generate a fixed batch of latent vectors (called fixed_noise) based on a Gaussian distribution to monitor the generator's learning curve. Throughout the training loop will continue until the noise has be reduced, this fixed_noise is regularly entered into (G), enabling us to see how the output images change over iterations.

F. Training DC- GAN

Training a Generative Adversarial Network (GAN) requires a multi-step process that carefully takes optimization strategies and hyperparameters into account. Mode collapse and other issues must be avoided by following advised processes and practices because of the inherent complexity of GAN.

The two main components of the training process they are generator and the discriminator. The first component is to update the discriminator such that it can accurately distinguish between real and the fake samples of the images. This can be achieved by building separate mini-batches for real and fake images, figuring out the loss for each batch, and then updating the discriminator's parameters using gradient descent.

➢ *Part-1 Train the Discriminator*

The main goal at this point is to properly train the discriminator to differentiate between the actual and bogus inputs. We use gradient ascent of stochastic to update the discriminator and follow Goodfellow's methodology to improve its performance. Our objective is to maximize the

function that assesses how well the discriminator separates authentic data from bogus data. We divide the procedure into two parts in order to do this.

First, a batch of actual dataset samples are created, run through the discriminator, and then the gradients are computed in a backward pass after determining the loss based on the sample's categorization as real or fake. Next, we use the current generator to create a batch of fictitious samples, which we then run through the discriminator, compute the loss, and accumulate gradients in a similar fashion. Through an optimization process, we adjust the discriminator's parameters by summing the gradients from the real and fictitious batches.

➢ *Part-2 Train the Generator*

In the next step, we focus on training the Generator network. The objective is to enhance the quality of generated fake samples by minimizing the loss function, specifically aiming to maximize the log probability of the Discriminator being fooled by these generated samples. Initially proposed by Goodfellow, this approach often lacks sufficient gradients, especially during early training stages. To address this, we adopt a strategy of maximizing the log probability of the Discriminator correctly classifying the generated samples as real. In our implementation, we achieve this by classifying the Generator's output from the previous step using the Discriminator, computing the Generator's loss based on real labels, computing gradients in a backward pass, and updating the Generator's parameters using an optimizer step. Using real labels for the loss function may seem odd, but it serves our goal by enabling us to take advantage of the BCE Loss's log(x) component. We also graphically track the Generator's performance at the end of each training epoch and present important statistics to track the progress of training. These data contain the average output of the discriminator for both real and fake batches, as well as discriminator and generator loss values.

For visually observing the G' training process, we will lastly perform some statistic reporting and pushing our fixed noise batch to the generator at the conclusion of each epoch. The following training statistics have been reported:

- Loss_D - discriminator loss computed as log(D(x)) + log(1-D(G(z))), which is the total of losses for all actual and all fake batches.
- Generator loss expressed as log(D(G(z))) is called Loss_G.
- D(x): the discriminator's average output (for the entire real batch) across the batch. When G improves, this should potentially converge to 0.5 from a starting point near Consider the reasons behind this.
- Note: Depending on how many epochs you run and whether you eliminated any data from the dataset, this phase may take some time.

IV. IMLEMENTAION

A. Dataset

For this study, we primarily used data from the University of California, Berkeley Segmentation Dataset and Benchmark 500 (BSDS500) datasets [18]. These datasets served as the foundation for the training of our denoising models for both color and grayscale images. Nevertheless, since these datasets are too small for effective network improvement, we enhanced the 500 color images from BSDS500 by adding Gaussian white noise with various standard deviations which results a dataset with 1500 noisy images.

Then, in order to train the color picture denoising model, we increased the size of our dataset by producing 3000 pairs of noisy, blurry images using methods like image flipping. To train the grayscale image denoising model, a final dataset of 2400 clear-noisy image pairs was obtained by applying the same augmentation to the 400 grayscale photos from BSDS500.

The standard colour set was used to evaluate the denoising performance for both colour and grayscale images. To ensure that our training and testing datasets remained distinct, we limited our denoising trials to the testing dataset. We mimicked noise observed in the real world by utilizing Gaussian noise, which offers a straightforward approximation for managing complex noise distributions. We also employed data augmentation to increase the model's generalization ability. Specially flipping technique which is used to improve the model's ability to gain spatial invariance across various image orientations.

B. Evaluation Metrics

➢ *Peak Signal-to-Noise Ratio (PSNR)*

PSNR is a performance evaluation metric which is used to assess image quality. It evaluates the denoising performance in the presence of ground truth, noise-free images. It matches pixel-to-pixel variations between the ground truth and denoised images. This is calculated using the mean square error (MSE) and it is given by the following formula:

$$\mathrm{MSE} = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} (X_{ij} - Y_{ij})^2$$

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{n^2}{\mathrm{MSE}(X,Y)}\right)$$

Where,

$M \times N$ are dimension of noisy images.

Xij and Yij represents to the pixel values.

$\boldsymbol{U}$ and j pixel coordinates inside the image.

n is the greatest grayscale level of the image.

➢ *Structural Similarity Index (SSIM)*

This method is also used to evaluate picture denoising, it is a metric that evaluates image similarity by taking into account numerous variables, such as contrast, brightness, and structure. Equations presents the SSIM calculating procedure.

$$\begin{cases} l(X,Y) = \dfrac{2\mu_X\mu_Y + C_1}{\mu_X^2 + \mu_Y^2 + C_1} \\ c(X,Y) = \dfrac{2\sigma_X\sigma_Y + C_2}{\sigma_X^2 + \sigma_Y^2 + C_2} \\ s(X,Y) = \dfrac{\sigma_{XY} + C_3}{\sigma_X\sigma_Y + C_3} \end{cases}$$

$$\mathrm{SSIM}(X,Y) = l(X,Y) \cdot c(X,Y) \cdot s(X,Y)$$

Where the two images in the structural similarity comparison are denoted by X and Y, and the similarity in brightness, contrast, and structure is indicated by $l(X,Y)$, $c(X,Y)$, and $s(X,Y)$, respectively. The two images' pixel means are represented by μX and μY, while their pixel standard deviations are represented by σX and σY. The covariance between the two images is represented by σXY. The constants $C11$, $C22$, and $C33$ prevent a zero denominator, thereby guaranteeing the validity of the structure. Under typical conditions, $C1=(K1\times L)1=(1\times 1)$, $C2=(K2\times L)2=(2\times 2)$, and $C3=(C2/2)3=(2/2)$ have the values $K11 = 0.01$, $K22 = 0.03$, and $L = 255$.

V. RESULTS AND DISCUSSION

A. Quantitative Analysis

Several denoising techniques are evaluated in this work using the BSD68 dataset with Gaussian white noise of different intensities (σ = 15, 25, and 50), including WGAN-VGG, Re-GAN, DnCNN, and BM3D. The experimental results show that, with significant gains in both PSNR and SSIM, our suggested technique continuously performs better than other algorithms at varying noise intensities. For example, our approach yields improvements in SSIM and an average increase in PSNR of 9.05 dB over BM3D at σ = 15. Likewise, when compared to previous methods, our technique significantly improves PSNR and SSIM at higher noise levels (σ = 25 and 50), demonstrating its strong denoising powers. Fig. 3 shows the comparison of denoising algorithms.

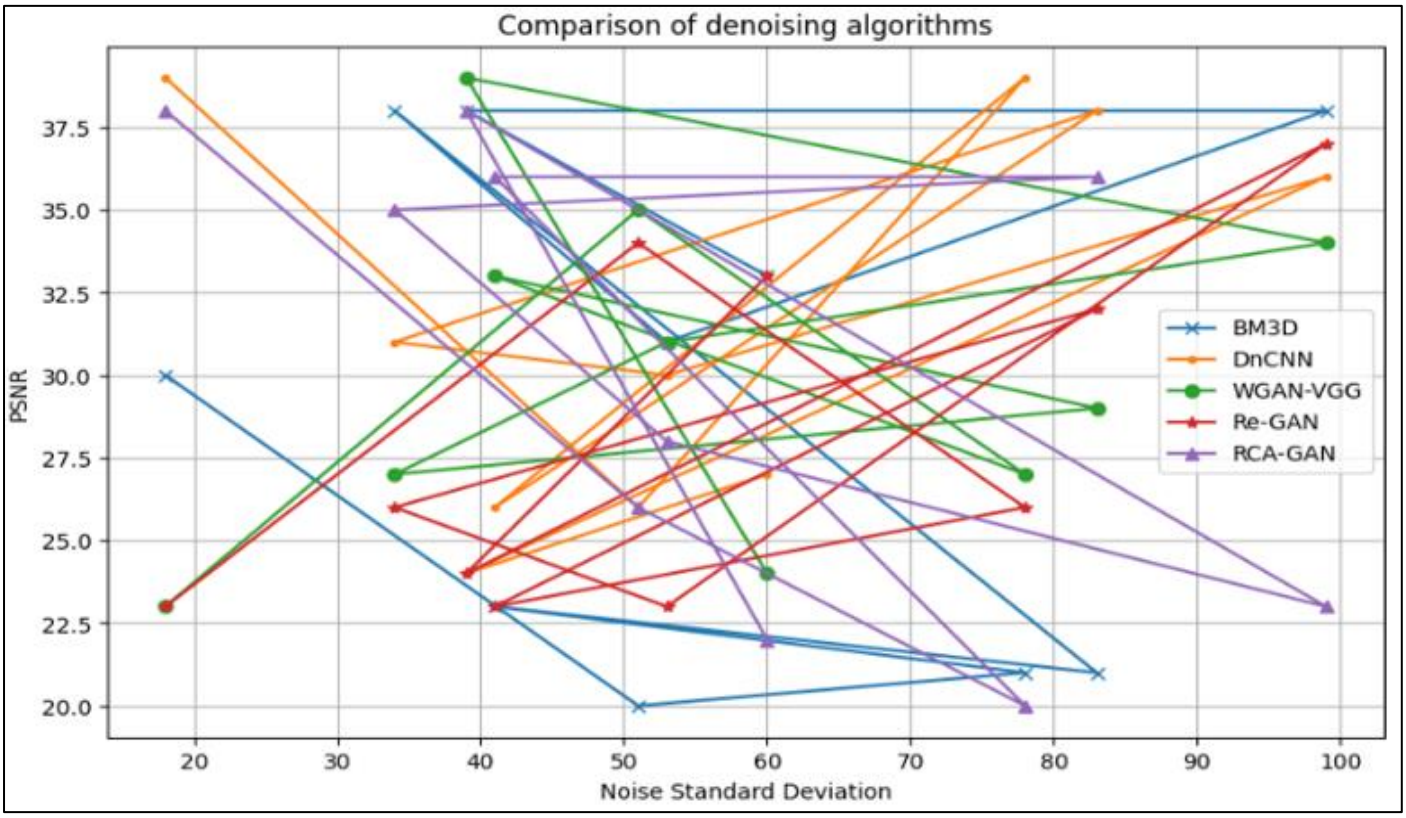

Fig 3 Comparison of Denoising Algorithms

B. Time Complexity Analysis

When assessing denoising methods, temporal complexity is just as important as conventional measurements like PSNR and SSIM. RCA-GAN, our algorithm, outperforms other techniques in terms of efficiency and significantly shortens the denoising time. It has a shorter average denoising time in a CPU context than WGAN-VGG, Re-GAN, BM3D, and DnCNN. It shows notable gains in denoising efficiency over DnCNN, WGAN-VGG, and Re-GAN in a GPU context. Our algorithm's attention mechanism, which maximizes feature use while lowering computational cost, is responsible for these gains. As a result, our technique achieves better runtime efficiency performance. Fig. 4 shows the performance analysis.

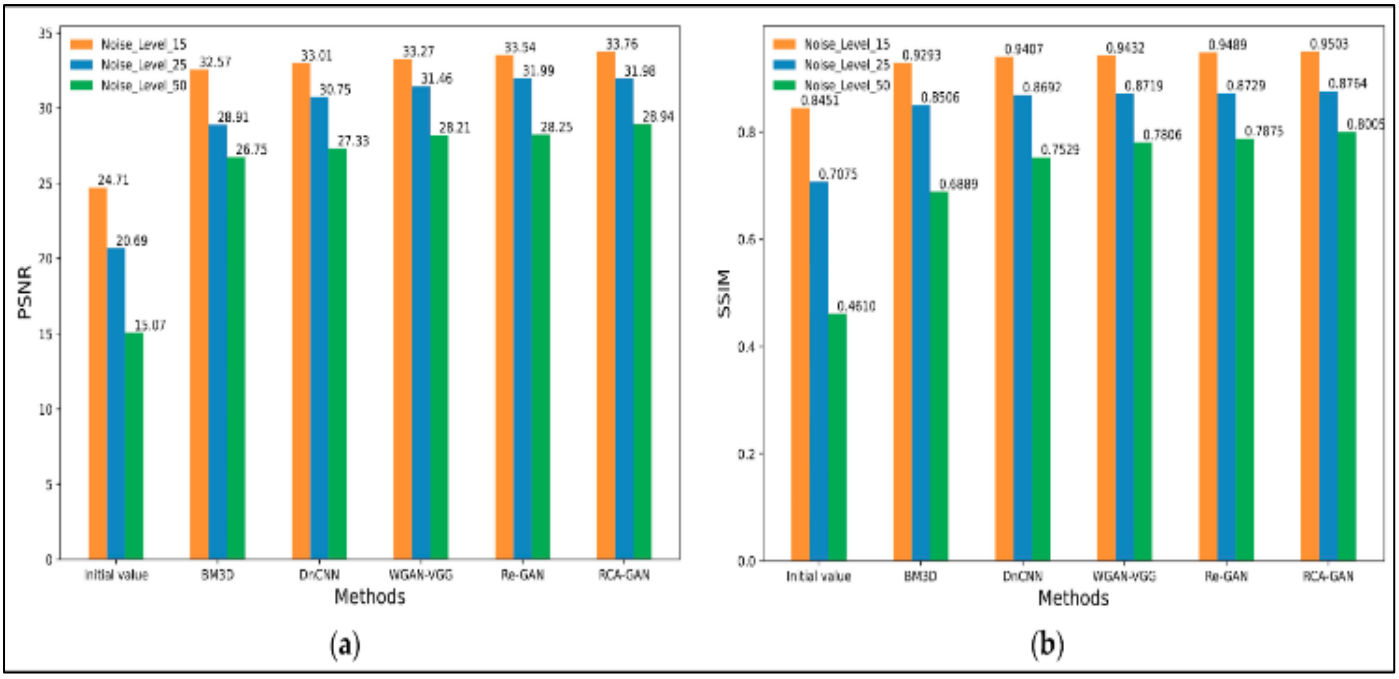

Fig 4 Performance Analysis of Denoising Methods

VI. CONCLUSION

This work introduces an improved GAN-based image denoising technique called RCA-WGAN, which addresses the issue of standard denoising algorithms losing edge and fine-grained information in denoised images. RCA-WGAN integrates cooperative attention mechanisms and residual structures in the feature extraction section of the generator network. For extracting more information from the image and reduce noise without compromising image quality, it also makes use of a global residual link. A multimodal loss function that has texture loss, adversarial loss, perceptual feature loss, and pixel space content loss is constructed using a weighted summation for optimizing noise reduction. Apart from this, the proposed denoising method minimizes texture loss caused by the denoising process by utilizing important RGB channel attributes.

The comparative analysis between the proposed technique and four widely-used denoising algorithms namely WGAN-VGG, DnCNN, Re-GAN, and BM3D demonstrates how well the former restores texture details in images. Experimental results of this study show that the proposed method performs exceptionally well in denoising, especially with improvements made to the denoising network module and loss function module, as measured by objective assessment criteria such as PSNR and SSIM values. When it comes to noise reduction without sacrificing image texture features, the proposed method performs better than other approaches. The denoising process still has a significant problem with handling complex noise in practical situations. Subsequent investigations will concentrate on enhancing the real-time processing and complex noise reduction capabilities of RCA-WGAN by means of further enhancements.

REFERENCES

[1]. Liu, Zhe, Wei Qi Yan, and Mee Loong Yang. "Image denoising based on a CNN model." *2018 4th International conference on control, automation and robotics (ICCAR)*. IEEE, 2018.

[2]. Lefkimmiatis, Stamatios. "Universal denoising networks: a novel CNN architecture for image denoising." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

[3]. Zhang, Kai, et al. "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising." *IEEE transactions on image processing* 26.7 (2017): 3142-3155.

[4]. P. D. S. Prasad, R. Tiwari, M. L. Saini, and Savita, "Digital Image Enhancement using Conventional Neural Network," 2023 2nd Int. Conf. Innov. Technol. INOCON 2023, 2023, doi: 10.1109/INOCON57975.2023.10100995.

[5]. J. Sarmah, M. L. Saini, A. Kumar, and V. Chasta, "Performance Analysis of Deep CNN, YOLO, and LeNet for Handwritten Digit Classification," *Lect. Notes Networks Syst.*, vol. 844, pp. 215–227, 2024, doi: 10.1007/978-981-99-8479-4_16.

[6]. ZhiPing, Qu, et al. "A new generative adversarial network for texture preserving image denoising." *2018 Eighth International Conference on Image Processing Theory, Tools and Applications (IPTA)*. IEEE, 2018.

[7]. Alsaiari, Abeer, et al. "Image denoising using a generative adversarial network." *2019 IEEE 2nd international conference on information and computer technologies (ICICT)*. IEEE, 2019.

[8]. Zhong, Yue, et al. "A generative adversarial network for image denoising." *Multimedia Tools and Applications* 79 (2020): 16517-16529.

[9]. Chen, Songkui, et al. "Image denoising with generative adversarial networks and its application to cell image enhancement." *IEEE Access* 8 (2020): 82819-82831.
[10]. Chen, Jingwen, et al. "Image blind denoising with generative adversarial network based noise modeling." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.
[11]. Li, Yuqin, et al. "A novel medical image denoising method based on conditional generative adversarial network." *Computational and Mathematical Methods in Medicine* 2021.1 (2021): 9974017.
[12]. Sheng, Zehua, et al. "Frequency-domain deep guided image denoising." *IEEE Transactions on Multimedia* 25 (2022): 6767-6781.
[13]. Zhang, Kai, Wangmeng Zuo, and Lei Zhang. "FFDNet: Toward a fast and flexible solution for CNN-based image denoising." *IEEE Transactions on Image Processing* 27.9 (2018): 4608-4622.
[14]. M. Sohail, M. Lal Saini, V. P. Singh, S. Dhir, and V. Patel, "A Comparative Study of Machine Learning and Deep Learning Algorithm for Handwritten Digit Recognition," *Proc. Int. Conf. Contemp. Comput. Informatics, IC3I 2023*, pp. 1283–1288, 2023, doi: 10.1109/IC3I59117.2023.10397956
[15]. M. Lal Saini, B. Tripathi, and M. S. Mirza, "Evaluating the Performance of Deep Learning Models in Handwritten Digit Recognition," *Proc. - Int. Conf. Technol. Adv. Comput. Sci. ICTACS 2023*, pp. 116–121, 2023, doi: 10.1109/ICTACS59847.2023.10390027.
[16]. M. L. Saini, R. S. Telikicharla, Mahadev, and D. C. Sati, "Handwritten English Script Recognition System Using CNN and LSTM," Proc. InC4 2024 - 2024 IEEE Int. Conf. Contemp. Comput. Commun., 2024, doi: 10.1109/InC460750.2024.10649099.
[17]. Y. Singh, M. Saini, and Savita, "Impact and Performance Analysis of Various Activation Functions for Classification Problems," *Proc. IEEE InC4 2023 - 2023 IEEE Int. Conf. Contemp. Comput. Commun.*, 2023, doi: 10.1109/InC457730.2023.10263129.
[18]. https://github.com/BIDS/BSDS500/tree/master/BSDS500/data/images