

StructGAN: Image Restoration Maintaining Structural Consistency Using A Two-Step Generative Adversarial Network

Authors

Nahian Muhtasim Zahin
160041008

Md. Mushfiqur Rahman
160041011

Kazi Raiyan Mahmud
160041058

Supervised by

Md. Hasanul Kabir, Ph.D.
Professor, Department of CSE, IUT

***A thesis submitted to the Department of CSE
in partial fulfillment of the requirements for the degree of
B.Sc. Engg. in Computer Science and Engineering***



**Department of Computer Science and Engineering
Islamic University of Technology (IUT)
Organization of Islamic Cooperation (OIC)
*Dhaka, Bangladesh***

March 2021

Originality Statement

We hereby declare that this submission is our own work and to the best of our knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at IUT or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom we have worked at IUT or elsewhere, is explicitly acknowledged in the thesis. The authors also declare that the intellectual content of this thesis is the product of their own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.

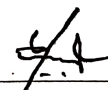
Authors:



Nahian Muhtasim Zahin
(Student ID: 160041008)

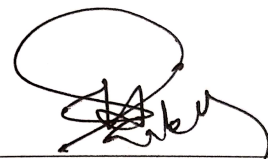


Md. Mushfiqur Rahman
(Student ID: 160041011)



Kazi Raiyan Mahmud
(Student ID: 160041058)

Supervisor:



Md. Hasanul Kabir, Ph.D.
Professor
Department of Computer Science and Engineering (CSE)
Islamic University of Technology (IUT)

04 March 2021

Abstract

Image restoration deals with the removal of noise, blurriness, missing patches, and other kinds of distortions in broken images. Traditional reconstruction and restoration approaches suffer from different kinds of limitations. In our work, we have improved upon those models by introducing novel structure loss that emphasizes the overall image structure rather than individual pixels. Our proposed model StructGAN can achieve a higher SSIM (Structural Similarity Index Measure) score while not massively compromising other noise metrics. Overall, our proposed model uses generative adversarial networks with a two-step generator network, a dual discriminator network, and coherent semantic attention (CSA) layer. The two-step generator helps refine the output. The dual discriminator ensures local and global correctness. The CSA layer ensures semantic consistency. Along with these, our model incorporates the novel structure loss. The structure loss is based on the Laplacian filter that calculates the overall structure-map of the image and tries to replicate the structure-map in the generation step. The results obtained by our model are qualitatively comparable to the performance of the state-of-the-art models. For certain metrics, e.g. SSIM, StructGAN quantitatively outperforms other models.

Acknowledgement

We are indebted to Professor Dr. Md. Hasanul Kabir for guiding us throughout this research. His valuable time and input were provided throughout this thesis work, from the initial phase of topic introduction, subject selection, hypothesis proposition, to the project implementation and finalization which helped us to do our thesis work correctly. Without his supervision, this research work would not have been possible.

We would also like to thank Mr. A. B. M. Ashikur Rahman, Mr. Redwan Karim Sony, Mr. Sabbir Ahmed, and all the other faculties of the Computer Vision Research Lab. Their time-to-time suggestions and interesting insights were massively helpful for our research. We are also grateful to Mr. Moshiur Farazi for his valuable opinion and advice regarding our work. His suggestions helped us improve our final work.

We would also like to thank the former head of department, Professor Dr. Muhammad Mahbub Alam, and the current head of department, Professor Dr. Abu Raihan Mostofa Kamal for creating a research-friendly environment at IUT. Such a research-friendly environment is vital for proper research.

Contents

Abstract	i
Acknowledgement	ii
Contents	iii
List of Figures	vi
List of Tables	viii
1 Introduction	1
1.1 Motivation	2
1.1.1 Super-resolution	2
1.1.2 In-painting	3
1.1.3 Denoising	3
1.1.4 Deblurring	4
1.2 Problem Statement	5
1.3 Objectives	5
1.4 Contributions	6
1.5 Organization of the thesis	6

2	Literature Review	7
2.1	Generative Models	7
2.1.1	Generative Adversarial Network	8
2.1.2	Image-to-image translation	9
2.2	Restoration Tasks and Methods	11
2.2.1	Statistical Modeling	11
2.2.2	Persistent Memory Modeling (PMM)	12
2.2.3	Denoising	12
2.2.4	Deblurring	13
2.2.5	Super Resolution	14
2.3	Loss Functions and Similarity Metrics	17
2.4	Image in-painting	18
2.5	Image in-painting with GANs	19
2.5.1	With two-step GAN	19
3	Proposed Methodology	21
3.1	Architecture Overview	21
3.2	Loss Functions	22
3.2.1	Adversarial Loss and Refinement Loss	23
3.2.2	Consistency Loss	23
3.2.3	Structure Loss	23
3.2.4	Combined Loss Function	25
3.3	Generator Networks	25
3.3.1	Rough Network	25
3.3.2	Refinement Network	27
3.4	Discriminator	29

3.4.1	Patch discriminator	29
3.4.2	Global discriminator	29
3.5	Adversarial training	30
4	Experimental Setup	31
4.1	Environment	31
4.2	Hyper-parameters	32
4.3	Dataset	33
4.3.1	Places 2 (Val)	33
4.3.2	COCO	33
4.4	Preprocessing	35
5	Result and Discussion	36
5.1	Sample Results	36
5.2	Quantitative Analysis	38
5.2.1	Ablation Study	38
5.2.2	Performance Evaluation	39
5.3	Qualitative Analysis	42
5.4	Discussion	42
6	Conclusion	43
6.1	Summary	43
6.2	Future Works	44
	References	46

List of Figures

1.1	Example of Image Super-Resolution [1]	2
1.2	Example of Image In-Painting [2]	3
1.3	Example of Image Denoising [3]	3
1.4	Example of Image Deblurring [4]	4
2.1	Simplified architecture of Generative Adversarial Networks (laptrinhx.com [5])	8
2.2	Simplified version of Pix2pix architecture (Isola et al. [6])	10
2.3	Examples of Image-to-image translation works (Isola et al. [6])	11
2.4	Free-Form Image Inpainting With Gated Convolution [7]	20
2.5	Coherent Semantic Attention for Image Inpainting [8]	20
3.1	Architecture overview of proposed model	22
3.2	Example of Laplacian Filter	24
3.3	Rough Network	26
3.4	Consistency loss (Similar to the work of Liu et al. [8])	28
3.5	Patch discriminator	30
4.1	Performance comparison between Google Colab and Kaggle	32
4.2	Few samples from the Places2 (Val) dataset	34
4.3	Few samples from the COCO dataset	34

4.4	Example of Center-square Occlusion Mask	35
5.1	Sample Outputs	36
5.2	Sample Outputs	37
5.3	Sample Outputs	38
5.4	SSIM Score of StructGAN	40
5.5	PSNR Score of StructGAN	41
5.6	MSE Score of StructGAN	41

List of Tables

5.1	Quantitative Analysis of our proposed model (StructGAN).	38
-----	--	----

Abbreviations

CNN	Convolutional Neural Network
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
GAN	Generative Adversarial Network
GPU	Graphics Processing Unit
MSE	Mean Squared Error
PSNR	Peak Signal-to-Noise Ratio
SSIM	Structural Similarity Index Measure
StructGAN	Structure GAN

Chapter 1

Introduction

Images, both physical and digital, can quite efficiently store visual information for a long period of time. All kinds of artworks, engravings on stones, photographs, etc., can be encompassed under the broader definition of the image. For ages, humans have been using this ingenious tool to pass information into the future. But images, especially physical images, do not last forever. With the passage of time, these go through different forms of wear and tear and get distorted. In the case of digitally stored images, wear and tear, over time, is not so common. But the poor quality of capturing devices or inefficiency of encryption algorithms can still cause deviation from the originally intended image. Noise, distortion, corruption, and any other form of deviation from the originally intended image results in loss of visual information. Generally, this loss is irreversible. Thus, no direct formula exists to reverse this process with full confidence. However, with some specific methods, this irreversible process can be reversed to some extent. Such methods are regarded as image restoration techniques. In plain words, image restoration is the task of regenerating the original image from the distorted image with some form of prior knowledge of the context of the image.

1.1 Motivation

Image restoration is a cognitive task. With a deep knowledge of the context of the original image and exceptional skill in generating images of similar form, an expert can quite successfully recreate an image with high accuracy. With enough practice, an amateur can improve their dexterity in recreating an almost identical copy of the original image, from only the distorted image. This paper deals with digital image restoration and considers it as a cognitive image-to-image translation task. Our primary motivation is to make a machine learning algorithm capable of restoring and reconstructing images.

Super-resolution, in-painting, denoising, and deblurring are some of the most commonly ventured aspects in digital image restoration literature. Our goal is to create a deep learning model that addresses all 4 of these topics simultaneously.

1.1.1 Super-resolution

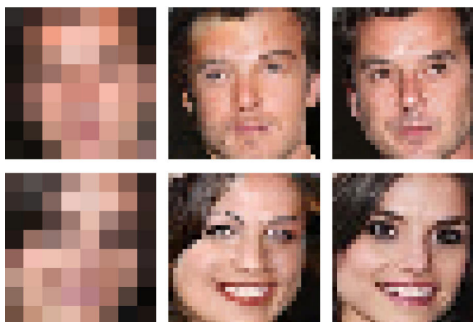


Figure 1.1: Example of Image Super-Resolution [1]

The process of up-scaling and enhancing the details within a low-resolution image is known as image super-resolution. In most cases, a low-resolution image is taken as the input of the system and the image is upscaled to a higher resolution to generate the output. The details in the high-resolution output are filled in where the details are essentially unknown.

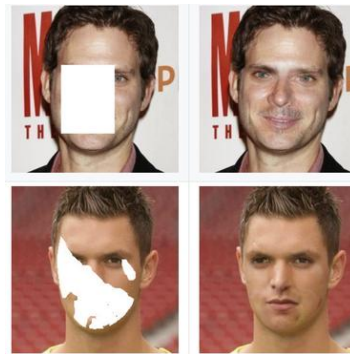


Figure 1.2: Example of Image In-Painting [2]

1.1.2 In-painting

Broken images often have large missing patches. Obtaining the original information for these missing portions is almost impossible. But with advanced algorithms and techniques and with prior knowledge of the image, it is possible to generate information for the missing portions that makes it consistent with the remaining image. This task of generating information for the missing portion of an image to make the overall image consistent is called image in-painting.

1.1.3 Denoising

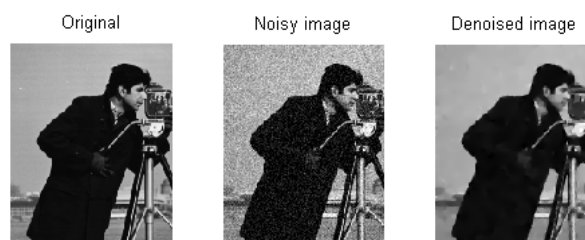


Figure 1.3: Example of Image Denoising [3]

The signal processing methods which can reconstruct a 1-D, 2-D, or 3-D signal from a noisy one is known as denoising. Its primary objective is to remove noise and retain useful information about the signal. In the case of images, there are different types of noises. A denoising technique detects and removes those noises. The biggest challenge here is, the

information obscured by noise is totally missing and is often irretrievable. So, the task of denoising involves generating new information coherent to the neighboring regions.

1.1.4 Deblurring



Figure 1.4: Example of Image Deblurring [4]

Blurry images do not have distinct edges. Deblurring algorithms aim at sharpening the edges and enhancing the structure of the image. Traditional image processing algorithms, like, morphological sharpening [9], un-sharp masking [10], etc., can successfully fix simple blurriness. However, these traditional algorithms often fail to deblur images with complex blurriness where a huge portion of original information goes missing. One such complex blurriness is the motion blur.

As exact regeneration is not possible, the success of a re-generator is measured by calculating the similarity of the generated image with the original non-distorted image. The primary objective of this research is to minimize the difference between X and X' and maximize their similarity.

$$\min(|X - X'|)$$

$$\max(\text{similarity}(X, X'))$$

The similarity is a broad and vague concept. It can be calculated in various ways and the performance of the re-generator vastly depends on the metric that is used to calculate the similarity. Traditionally, pixel-by-pixel mapping has been used for this purpose. If an exact copy of the image was necessary, then such a similarity metric could be of great use. But since image restoration focuses more on regenerating a semantically sensible version

of the image and not just a digital replica with inconsistent information, only using the pixel-by-pixel similarity is counter-intuitive.

1.2 Problem Statement

If an original image, represented by X , has a noise-map N . And the noise in the image has an intensity α , then the distorted image can be represented by:

$$f(X) = \alpha N + (1 - \alpha)X$$

In this research, the primary goal is to obtain the original image, X , when the distorted image, $f(X)$, is given. Since N and α are totally unknown and can be anything, it ultimately requires an intelligent system to find those missing information and generate X' from some prior knowledge of the image, the noise, and the environment.

$$X' = f(\text{Distorted}, \text{prior})$$

Therefore, image restoration and reconstruction can be defined as the task of creating a function that uses a distorted image and utilizes the image-prior to generate an output very similar to the original image.

1.3 Objectives

The primary objective of this research is to find an algorithm that can generate missing portions of an image while maintaining the overall structural consistency. Regeneration tasks are quantitatively compared with different scoring metrics. The additional objective of this research is to have high scores in these metrics.

1.4 Contributions

In this report, we have introduced a novel approach to calculate the structural similarity of images in in-painting tasks. Our approach focuses more on the overall structure of the image rather than individual pixels. We have incorporated this mechanism in the form of a loss function in our two-step generative model. Our model achieves a high SSIM score.

1.5 Organization of the thesis

This report has 6 chapters. These are: Introduction, Literature Review, Proposed Methodology, Experimental Setup, Result and Discussion, and Conclusion. The Introduction chapter introduces the problem statement and gives an overview of the report. The Literature Review chapter discusses related works in the domain. The Proposed Methodology chapter describes our proposed model. The Experimental Setup chapter gives a detailed explanation of our experiments. The Result and Discussion chapter gives a qualitative and a quantitative analysis of the outputs of our work. The Conclusion chapter concludes the work with a brief summary of our work. Prior to these 6 chapters, this report has a list of all the figures, a list of all the tables, and a list of all the abbreviations used in this report.

Chapter 2

Literature Review

Much work has been done on image restoration. But contrary to our approach, most of these papers only target one form of distortion, whereas, our goal is to create an algorithm capable of solving multiple types of image distortion.

2.1 Generative Models

The use of deep learning has shown promising results when it came to discovering models capturing the probability distribution of different types of data like images, audios, natural languages, etc. Most of these models were discriminative models. As a whole, these are known as Auto-Encoders. These use the Kullback-Leibler(KL) divergence. Although they work well for some cases, they massively fall short in the case of a true generation where samples are very divergent from generated image. This divergence causes the generative loss to bloat up.

2.1.1 Generative Adversarial Network

Producing good generative models has not been much success in the past due to the difficulties of calculating probabilities and utilizing the benefits of piece-wise linear functions. Ian Goodfellow et al. [11] proposed a framework that can produce generative models with very good accuracy. The basis of this architecture is the Jensen-Shannon (JS) divergence instead of the KL divergence. This new framework has pushed the boundaries of deep learning and has provided some handy solutions to a lot of problems. In this paper, two models are trained simultaneously. One model is the generative model (G) which works with the data distribution and generates new data. The other one is a discriminative model (D) which calculates the probability that an input sample came from the training data rather than the other model, G. The discriminative model (D) takes data as an input and outputs a scalar value, either real or fake. The goal of the generative model (G) is to fool the other model by producing such data which maximizes the probability of D classifying a fake input as real. On the other hand, the goal of the discriminative model (D) is to correctly classify any input data fed into it. This adversarial process makes both the models better as they basically compete against each other.

A simplified architecture of Generative Adversarial Networks has been shown in Figure 2.1.

GAN Architecture

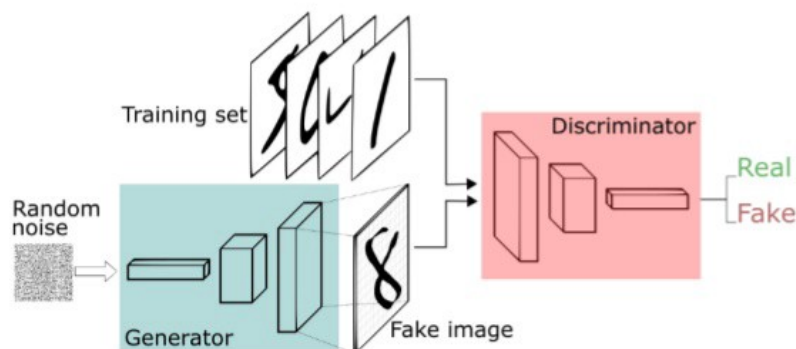


Figure 2.1: Simplified architecture of Generative Adversarial Networks (laptrinhx.com [5])

This concept opened new dimensions in the field of deep learning. A lot of other variations of GAN architecture were later developed which could also produce satisfactory results. The whole idea of this paper is highly relevant to our research. A major portion of our implementation incorporated the core scheme of this architecture.

2.1.2 Image-to-image translation

An image can be represented in many ways like RGB representation, edge map, gradient field, semantic map. When working with digital images, we need to work with a variety of these representations according to our purpose. Isola et al. [6] mainly focuses on image-image translation using **Conditional Adversarial Networks**. Zhu et al. [12] also use image-to-image translation but for unpaired images.

The goal of this network is firstly to learn a mapping from an input image to an output image. Secondly, the network also learns a loss function which trains this model to achieve a more general approach to this problem. If the same job was approached more traditionally with CNNs, the loss function for each type of input would vary and had to be designed manually which is more challenging and less efficient. As discussed earlier, GANs are generative models that learn a mapping from random noise vector z to output image y , $G:z \rightarrow y$. In contrast, the cGANs learn to map y from observed image x and random noise vector z . $G:(x, z) \rightarrow y$. For building this network, the “U-net” architecture was used. In the “U-net” architecture, the input is first downsampled until a point that is known as the bottleneck. Then the whole process is reversed. To ensure that some low-level information is not lost during the downsampling process, skip connections are used between mirrored layers across the whole architecture. These skip connections pass necessary low-level information between layers.

Figure 2.2 illustrates a simplified version of the training architecture of Pix2pix proposed by Isola et al. [6]

It was seen that common losses like the **L1** and **L2** norms, can accurately capture low-

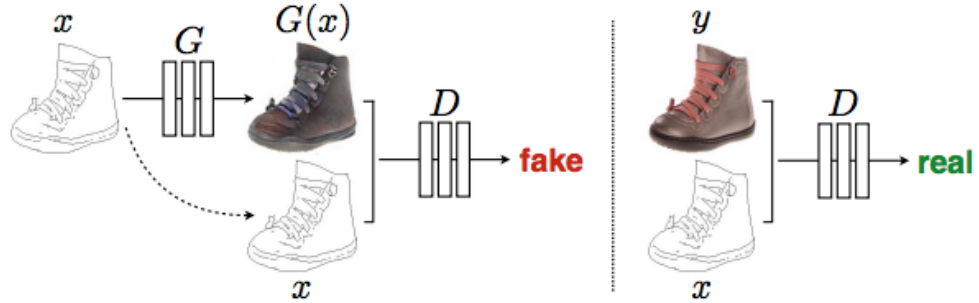


Figure 2.2: Simplified version of Pix2pix architecture (Isola et al. [6])

level frequency information of an image though they failed to do so with the high-level frequencies. As a result, the images produced were blurry. The authors of this paper smartly added the L1 loss with the loss function instead of building a whole new framework. This worked well for both the low and high-level frequency information of any input image and produced good results.

The method of this paper was tested on numerous tasks and datasets like:

- Semantic segmentation map \longleftrightarrow realistic photo
- Architectural facade segmentation \longrightarrow realistic photo
- Black and white \longrightarrow colored photos
- Map \longleftrightarrow aerial photo
- Edges \longrightarrow photo
- Sketch \longrightarrow photo
- Day \longrightarrow night
- Thermal \longrightarrow color
- Photo with missing pixels \longrightarrow in-painted photo

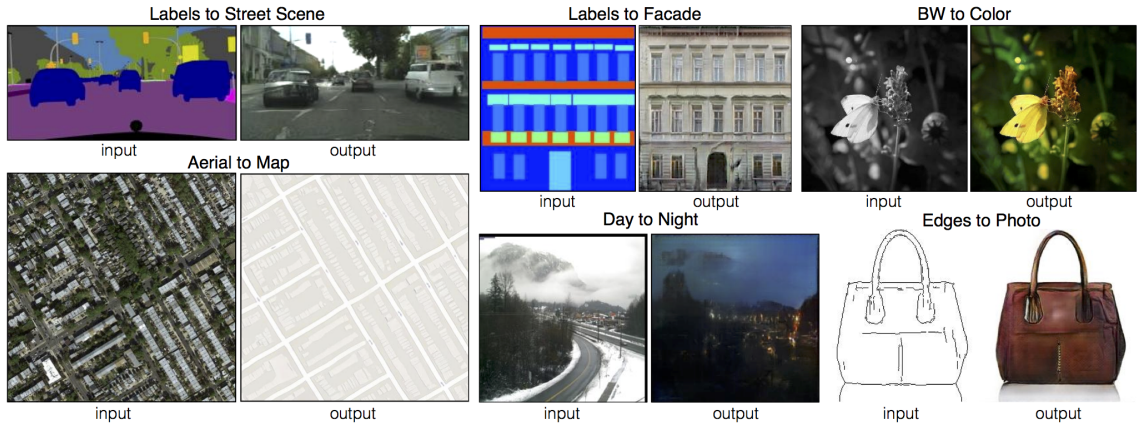


Figure 2.3: Examples of Image-to-image translation works (Isola et al. [6])

Few examples of image-to-image translation domains have been illustrated in Figure 2.3.

All of the results showed that their loss function was working better than any common or fixed loss function. This paper was pertinent to our research as they incorporated a new loss function with GAN architecture and also worked with some tasks (like image in-painting) which were similar to what we are trying to accomplish.

2.2 Restoration Tasks and Methods

2.2.1 Statistical Modeling

The use of statistical algorithms is also quite common in image restoration tasks. Zhang et al. [13] propose a method that statistically characterizes local smoothness and nonlocal self-similarity of natural images to handle image restoration.

Most papers on image restoration assume images to be locally smooth except for the edges. Regularization techniques based on this assumption (total variation (TV), half quadrature formulation, and Mumford-Shah (MS) models) can preserve edge smoothness effectively but smear out image details. The alternative to this assumption is the use of NLM (non-local means) in creating weighted filters by analyzing surrounding pixels from the image prior. In recent literature, the use of non-local self-similarity property can be seen both in

pixel level (for denoising) and in block/patch level (for super-resolution and deblurring).

The paper proposes a novel model (Joint Statistical Modeling) that combines the local statistical modeling in space-domain(2D) and non-local self-similarity in transform-domain (3D). The proposed regularization term is as follows:

$$\Psi_{JSM}(u) = \tau \cdot \Psi_{LSM}(u) + \lambda \cdot \Psi_{NLSM}(u) \quad (2.1)$$

Here, Ψ represents regularization.

2.2.2 Persistent Memory Modeling (PMM)

A very common problem in deep CNNs is that the prior states/layers have very little influence on the subsequent ones. Tai et al. [14] propose a very deep persistent memory network (MemNet) that introduces a memory block, consisting of a recursive unit and a gate unit. The representations and the outputs from the previous memory block are sent to the gate unit which controls how much of the previous states should be reserved and how much of the current states will be added to the memory. MemNet addressed three image restoration tasks – image denoising, super-resolution, and JPEG deblocking. They used the following loss function:

$$\mathcal{L}(\Theta) = \frac{1}{2N} \sum_{i=1}^N \left| (\tilde{x})^{(i)} - D(\tilde{x})^{(i)} \right|^2 \quad (2.2)$$

2.2.3 Denoising

Image denoising is one of the most widely studied topics of computer vision. Many great works are available in this field. Their results are mostly excellent. Scientists have used a wide variety of algorithms for image denoising. The use of the statistical approaches [15–18], the non-local approaches [19–22], and the filtering approaches [23, 24] have shown the best results so far.

The statistical approaches mostly work with wavelet coefficients. Mihcak et al. [15] introduced a simple spatially adaptive statistical model for wavelet image coefficients and applied it to image denoising. Their model was inspired by another wavelet image compression algorithm, the Estimation Quantization Coder [25]. They modeled the wavelet image coefficients as zero-mean Gaussian random variables [26] with high local correlation. Their model presupposed a marginal prior distribution on wavelet coefficient variances. This distribution was estimated using the Maximum A Posteriori Probability rule. Then they applied an approximate Minimum Mean Squared Error estimation procedure to restore the noisy wavelet image coefficients. Despite the simplicity of their method, both in its concept and implementation, their denoising results are among the best reported in the literature.

Buades, Coll, and Morel [19] proposed a new measure, the method noise, to evaluate and compare the performance of digital image denoising methods. They firstly computed and analyzed this method for a wide class of denoising algorithms, namely, the local smoothing filters. Secondly, they proposed a new algorithm, the non-local means (NL-means), based on a non-local averaging of all pixels in the image. Finally, they presented some experiments comparing the NL-means algorithm and the local smoothing filters.

The results obtained by the aforementioned state-of-the-art techniques are already great so our research will not delve deeper into them. But we have plans to leverage their techniques and incorporate them into our system.

2.2.4 Deblurring

Prior works on deblurring show promising results. Most deblurring literature deal with motion blur [27–32]. In our research, we are also primarily targeting motion blur. Shan et al. [32] presented a novel algorithm for removing motion blur from a single image. Their method constructs a deblurred image on the basis of a probabilistic model. The probabilistic model that they use, unifies blur kernel estimation with unblurred image restoration. They presented a thorough analysis of the common reasons for artifacts found in outputs

of current deblurring methods. They also introduced several novel terms within their probabilistic model. These terms include a) a model of the spatial randomness of noise in the blurred image and b) a new local smoothness prior that reduced the ringing effect of artifacts by constraining contrast in the unblurred image wherever the blurred image has low contrast. Finally, they described an efficient optimization scheme that alternates between blur kernel estimation and unblurred image restoration until convergence. As a result of these steps, they were able to produce high-quality deblurred results in low computation time.

2.2.5 Super Resolution

Super-resolution has made huge progress in recent years. Previously, cubic and bicubic methods were primarily used to zoom-in or expand images. But with modern technologies and algorithms, currently, deep learning is massively used in the image super-resolution domain. Sahu [33] quite brilliantly explains the evolution of the use of deep learning in a single image super-resolution domain.

2.2.5.1 Interpolation

Prior to the wide use of deep learning techniques, interpolation was the go-to method for researchers working in the super-resolution domain. Common interpolation methods are:

- Nearest-neighborhood interpolation
- Bilinear interpolation
- Bicubic interpolation

2.2.5.2 SRCNN

After the success of fully convolutional neural networks (FCNN) [34], its popularity in various fields of computer vision promulgated rapidly. CNNs has two primary functional blocks – one extracts features and the other one classifies outputs. The fully-connected (FC) layers at the end-part of CNNs are the classifier whose task is to map the extracted features to class probabilities. SRCNN [35] was one of the primary applications of FCNN. In SRCNN, first, the image is unsampled using any interpolation technique (Dong et al. [35] recommended bicubic interpolation). The output of the interpolation is fed into a simple FCNN. The method does not use any pooling operation, so the output has the same spatial size as that of the unsampled input image. In the last step, SRCNN computes the MSE loss between the target high-resolution image and the obtained output.

2.2.5.3 SRResNet and Sub-pixel convolution

With the promising progress of SRCNN, the next step in the evolution was achieved through the use of ResNet (CNN architecture with skip connections) in super-resolution models. SRResNet [36] replaced simple convolutional blocks of SRCNN with residual blocks. This gave a huge boost in the accuracy of the algorithm.

Upsampling operations were implemented with stridden convolution gradients which adds zero values to upscale the image, which has to be later filled in with meaningful values.

2.2.5.4 Perceptual Loss

MSE or MSE-based error mechanisms only measure the pixel-difference between two corresponding pixels in the generated image and the ground-truth image. These are generally too smooth and thus have poor perceptual quality. Therefore, it is advised to not check PSNR alone while comparing the performance of any two methods in such tasks [33].

Perceptual loss [37] is calculated by finding the changes between two images based on

high-level representations from a pre-trained CNN model. The feature is used to compare high-level variations between pictures, such as content and style differences. In our work, we have utilized this technique to some extent.

2.2.5.5 SR-GAN

The GAN technique allows reconstructions to move into search space regions with a high likelihood of containing photo-realistic images, taking them closer to the natural image.

SR-GAN [36] is another GAN-based network. Upsampling is done by the Generator using ResNet and sub-pixel convolution.

$$\min_{\theta_G} \max_{\theta_D} \left(E \left[\log D_{\theta_D}(I^{HR}) \right] + E \left[\log \left(1 - D_{\theta_D}(G_{\theta_D} I^{LR}) \right) \right] \right)$$

$$l_{Gen}^{SR} = \sum_{n=1}^N -\log D_{\theta_D} \left(G_{\theta_G} \left(I^{LR} \right) \right)$$

$$l^{SR} = l_X^{SR} + 10^{-3} l_{Gen}^{SR}$$

Here, the discriminator tries to maximize the net loss and the generator tries to minimize it to minimize it. Though SR-GAN has great results, the hallucinated details are often accompanied by unpleasant artifacts.

2.2.5.6 ESRGAN

The latest addition to the SR algorithms is the Enhanced Super-Resolution Generative Adversarial Network (ESRGAN) [38]. It is capable of generating realistic textures better than all the previously mentioned algorithms. ESRGAN improved and enhanced all the key components of SR-GAN – network architecture, adversarial loss, and perceptual loss.

In particular, the paper introduced a novel type of dense block, namely, Residual-in-Residual Dense Block. The block does not have any batch normalization layer. Their other contribution was the use of relativistic GAN for discriminators where the discriminator predicts relative realness instead of absolute realness. Finally, they improved perceptual loss by using the features before activation, which could provide stronger supervision for brightness consistency and texture recovery.

2.3 Loss Functions and Similarity Metrics

Zhao et al. [39] evaluates the performance of L2 loss for different image restoration tasks (image super-resolution, JPEG artifacts removal, and joint denoising plus demosaicking) and proposes a novel loss function that works better for image restoration. They compared the L2 loss function with four other image quality error metrics: L1, SSIM (structural similarity index), MS-SSIM (multi-scale structural similarity index) [22] and mix (a novel loss the paper proposes). The paper claims, the mixed loss works better because the human perception of image quality does not resonate with L2 loss but with SSIM and MS-SSIM. The paper uses the same CNN model and applies different loss functions to it to obtain comparable outputs. The comparison shows that L1 loss alone works better than SSIM and MS-SSIM. But their proposed mixed loss performs better than both L1 and SSIM/MS-SSIM losses.

Patch Loss:

$$\mathcal{L}^\varepsilon(P) = \frac{1}{N} \sum_{p \in P} \varepsilon(p) \quad (2.3)$$

L1 Loss:

$$\mathcal{L}^{l_1}(P) = \frac{1}{N} \sum_{p \in P} |x(p) - y(p)| \quad (2.4)$$

SSIM Function:

$$\text{SSIM}(p) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_2} \cdot \frac{2\sigma_x\sigma_y + C_1}{\sigma_x^2 + \sigma_y^2 + C_2} \quad (2.5)$$

$$= l(p) \cdot cs(p) \quad (2.6)$$

SSIM Loss:

$$\mathcal{L}^{SSIM}(P) = \frac{1}{N} \sum_{p \in P} 1 - \text{SSIM}(p) \quad (2.7)$$

$$\mathcal{L}^{SSIM}(P) = 1 - \text{SSIM}(\tilde{p}) \quad (2.8)$$

MS-SSIM function:

$$\text{MS-SSIM}(p) = l_M^\alpha(p) \cdot \prod_{j=1}^M cs_j^{\beta_j}(p) \quad (2.9)$$

MS-SSIM loss:

$$\mathcal{L}^{MS-SSIM}(P) = 1 - \text{MS-SSIM}(\tilde{p}) \quad (2.10)$$

Mix Loss:

$$\mathcal{L}^{Mix} = \alpha \mathcal{L}^{MS-SSIM} + (1 - \alpha) \cdot G_{\sigma_G^M} \cdot \mathcal{L}^{l_1} \quad (2.11)$$

2.4 Image in-painting

One of the most popular methods of reconstructing broken or damaged images is image in-painting. The damaged, deteriorating, or missing parts of images or artworks are filled in with the help of a neural network. In fact, machine-generated images in paints are better at filling up the missing parts than a human artist. To learn about in-painting, we need to discuss context encoders. It is a type of autoencoder that consists of an encoder, bottleneck, and decoder. Its purpose is to reduce the image size ignoring the noise of that image. Now the context encoder is a type of convolutional neural network which considers the surrounding of an image to predict the missing parts of that image. The encoder

part's duty is to try to capture the context of the image in a compact latent feature representation. The decoder, on the other hand, uses that representation to produce the missing image content. We feed our model with a huge dataset of images with missing bits. There are several ways to create the blocked part of an image, it is known as region mask. We can handle it in 3 ways;

- Central region: The central part of the image is set to zero. This is way too simple and causes generalization.
- Random block: Instead of putting the block in middle, it is randomized. Several overlapping squares take up-to one-fourth of the image.
- Random region: This creates sharp boundaries of the mask with arbitrary shapes around the image.

In painting, the model consists of an encoder, a decoder that works as the generator. This generates our desired image and tries to get better with the help of a discriminator. The discriminator finally with the help of the sigmoid function gives us a scalar output to decide how well the model did.

2.5 Image in-painting with GANs

All the aforementioned methods can fall under the broad domain of image restoration. But there are papers that target image restoration as a whole instead of the smaller domains.

2.5.1 With two-step GAN

Yu et al. [7] proposed a two-step generative model that roughly in-paints in the first step and refines the outputs in the next step. The model has that two-step GANs or gated GANs can give great performance in image restoration domain. Beside using two back-to-back generators, the also introduce two simultaneous discriminators. In Figure 2.4, the

overview of their proposed model has been described. Many other literature has developed upon this work and has added different kinds of attention mechanisms to it.

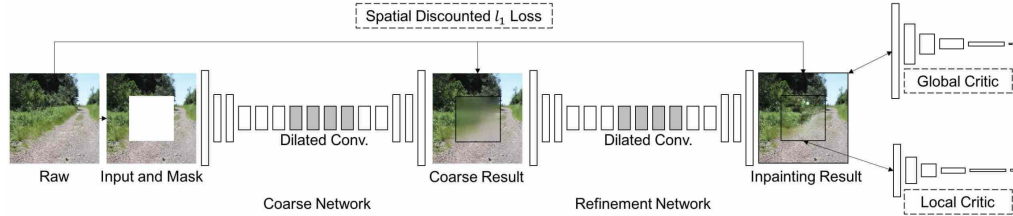


Figure 2.4: Free-Form Image Inpainting With Gated Convolution [7]

Liu et al. [8] proposed a special type of attention layer that is semantically coherent. Their paper uses the attention layer on top of the two-step GAN proposed by Yu et al. [7]. This improves the model even more. This model can capture semantic information and retain consistency across different parts of the image. The Figure 2.5 gives an illustration of the working mechanism of their model.

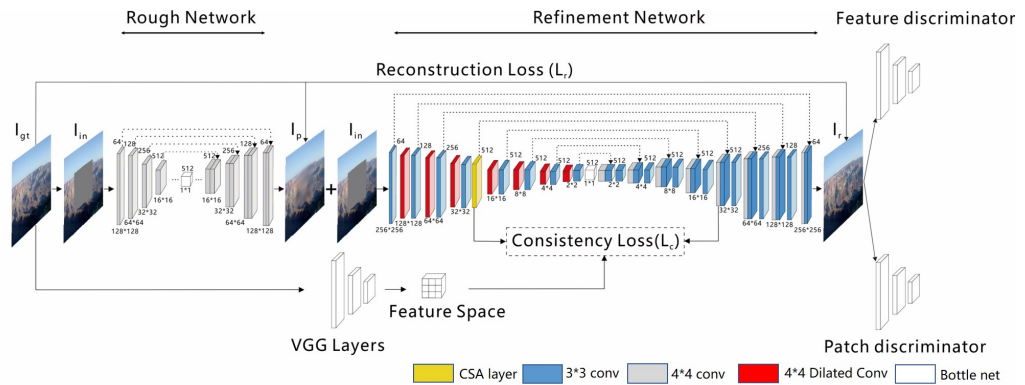


Figure 2.5: Coherent Semantic Attention for Image Inpainting [8]

The issue with structural consistency still persists in these works.

Chapter 3

Proposed Methodology

This paper treats image restoration as an image-to-image translation problem. Isola et al. proposes the conditional GAN (cGAN) [6] for image-to-image translation problems. In this paper, we have used a variation of cGAN to restore images.

3.1 Architecture Overview

The core architecture of our work is based on the model proposed by Liu et al. [8]. The model consists of two separate generators – rough network, and refinement network. These two generators are designed to perform two distinct tasks. The model also includes two discriminators. Inspired by Liu et al.’s work, we have also used a Coherent Semantic Attention (CSA) block in our refinement network. Our contribution to the architecture is the novel structure extractor sub-model and the structure loss obtained from this sub-model.

An overview of our proposed model has been given in Figure 3.1. The input to the model is marked with 'B Real'. After adding some sort of distortion, the image becomes 'A'. The goal of the model would be to produce an image almost similar to 'B real' from 'A'. Passing 'A' through the first u-net, 'B1 Fake' is obtained. Conditioning 'B1 Fake' on 'A',

we get the input for the next u-net. In this report, the first u-net is defined as Rough Network and the second u-net is defined as Refinement Network.

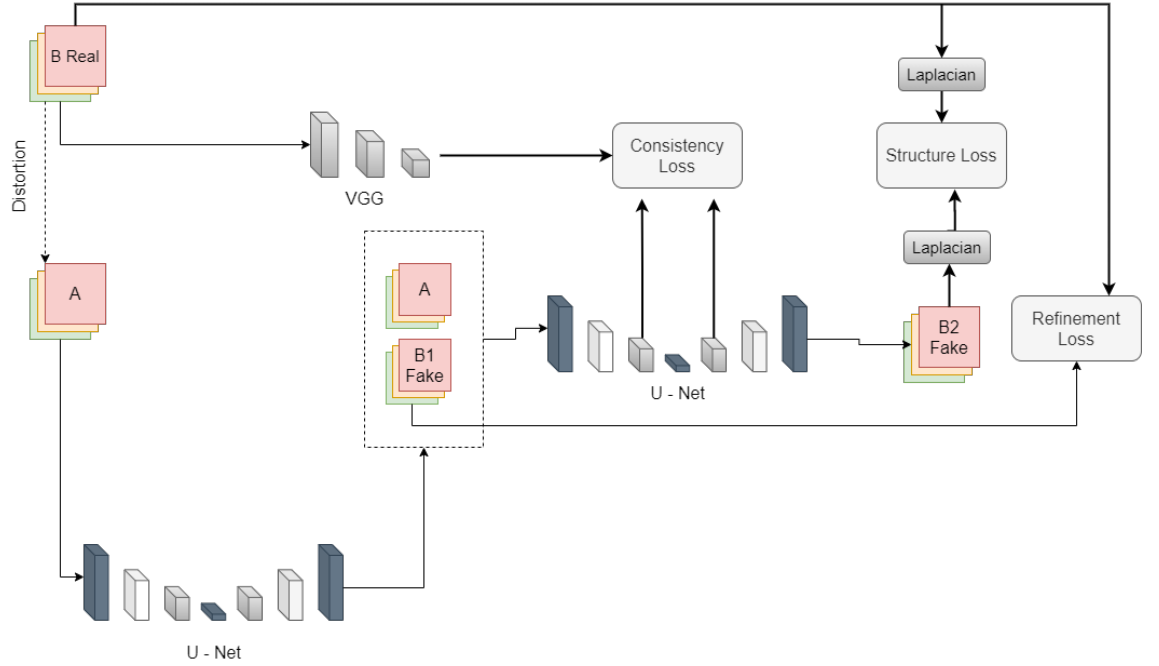


Figure 3.1: Architecture overview of proposed model

3.2 Loss Functions

Let for any example case, X be the distorted image and y be the ground-truth. The goal of the overall architecture would be to generate a generator function, such that, $G(X) \approx X'$. So, if $\mathcal{L}(X, y)$ gives the pixel-wise loss of the generated image and the original image. But for generative models, a pixel-by-pixel loss is not sufficient and a loss calculated directly does not ensure a good generative mechanism. So, this paper uses a combination of three other kinds of loss functions to overcome these problems.

3.2.1 Adversarial Loss and Refinement Loss

Adversarial loss is the min-max approach from game theory. This paper uses a variation of the adversarial loss similar to that of cGAN [6]. This adversarial loss facilitates the primary generative mechanism of the model by introducing a discriminator. Besides X and y , adversarial loss uses an additional noise vector z . The aim is to generate $G : x, z \rightarrow y$. The discriminator predicts how close the generated output is to x .

$$\mathcal{L}_{cGAN}(G, D) = E_y [\log D(y)] + E_{x,z} [\log(1 - D(G(x, z)))] \quad (3.1)$$

$$\mathcal{L}_{L1}(G) = E_{x,y,z} [y - G(x, z)] \quad (3.2)$$

The objective of the adversarial loss function is:

$$\mathcal{L}_r = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G) \quad (3.3)$$

3.2.2 Consistency Loss

Similar to the CSA inpainting paper [8], our model has consistency loss. This is a re-designed form of the perceptual loss. The loss is defined as:

$$\mathcal{L}_c = \sum_{y \in M} \|CSA(I_{ip})_y - \Phi_n(I_{gt})_y\|_2^2 + \|CSA_d(I_{ip})_y - \Phi_n(I_{gt})_y\|_2^2 \quad (3.4)$$

3.2.3 Structure Loss

The adversarial loss is sufficient in ensuring that the generator learns. But due to the pixel-wise L1 loss in the objective, the generator learns to get every pixel correct. However, in reality, the shape and structure of the objects are more important than pixel-wise correctness. An image with a different brightness level and color contrast is deemed

accurate if the overall structure matches. To achieve structural similarity, we have patch-wise compared the edge map of the generated image with that of the original image. So, besides minimizing the adversarial loss, the model has to minimize the structural differences.

Any function, $S : x \rightarrow edge(x)$, can be used for the filter. We have tried experimenting with first derivative filters, e.g., Sobel Filter, and second derivative filters, e.g, Laplacian Filter. Though the high noise-sensitivity is a big issue for Laplacian Filters, the early stages of experimentation show better result for the Laplacian Filter. Therefore our model uses a Laplacian operator-based edge detector. [40].

$$\Delta f_L = \frac{\delta^2 f}{\delta x^2} + \frac{\delta^2 f}{\delta y^2} + \frac{\delta^2 f}{\delta z^2} \quad (3.5)$$

This entire function can be achieved using the filter:
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$



Original Image



Image after Laplacian Filter

Figure 3.2: Example of Laplacian Filter

The use of a novel structure loss in addition to the adversarial loss is the primary contribution of this paper. The loss function objective is minimizing the difference of input and output structure:

$$\mathcal{L}_s = \frac{1}{n} \sum_{i=1}^n (f_L(G(X)) - f_L(y))^2 \quad (3.6)$$

3.2.4 Combined Loss Function

$$\mathcal{L} = \alpha(\beta\mathcal{L}_s + (1 - \beta)\mathcal{L}_{cGAN}(G, D)) + \lambda\mathcal{L}_{L1}(G) + (1 - \alpha - \lambda)\mathcal{L}_c \quad (3.7)$$

In Eq. 3.7, \mathcal{L}_{cGAN} , \mathcal{L}_c , and \mathcal{L}_s are refinement loss, consistency loss and structural loss respectively. α, β , and γ determines their influence in the overall loss.

3.3 Generator Networks

Our work heavily relies on the generator network. Like most GAN architectures, our work uses u-net in the generator network. Instead of using only one u-net, our model uses two u-net architectures. In this report, the first u-net is defined as Rough Network and the second u-net is defined as Refinement Network.

3.3.1 Rough Network

The rough in-painting network takes input image (distorted image) of size $3 \times 256 \times 256$. Like any generator network, this network has two portions – an encoder and a decoder. The key structural features of our rough network are:

- **Encoder:** The encoder consists of 4×4 convolution blocks. After each convolution block, the image size halves.
- **Decoder:** The decoder consists of subsequent deconvolution blocks. Skip-connection is added to each layer in the decoder portion from each corresponding encoder layer.
- **Loss:** L_1 reconstruction loss is used to train the rough network

The Figure 3.3 shows the architecture of the rough network. In the figure, the green boxes represent outputs of convolutional blocks and the red boxes represent the outputs of the deconvolutional blocks. There are skip connections between the conv and deconv blocks.

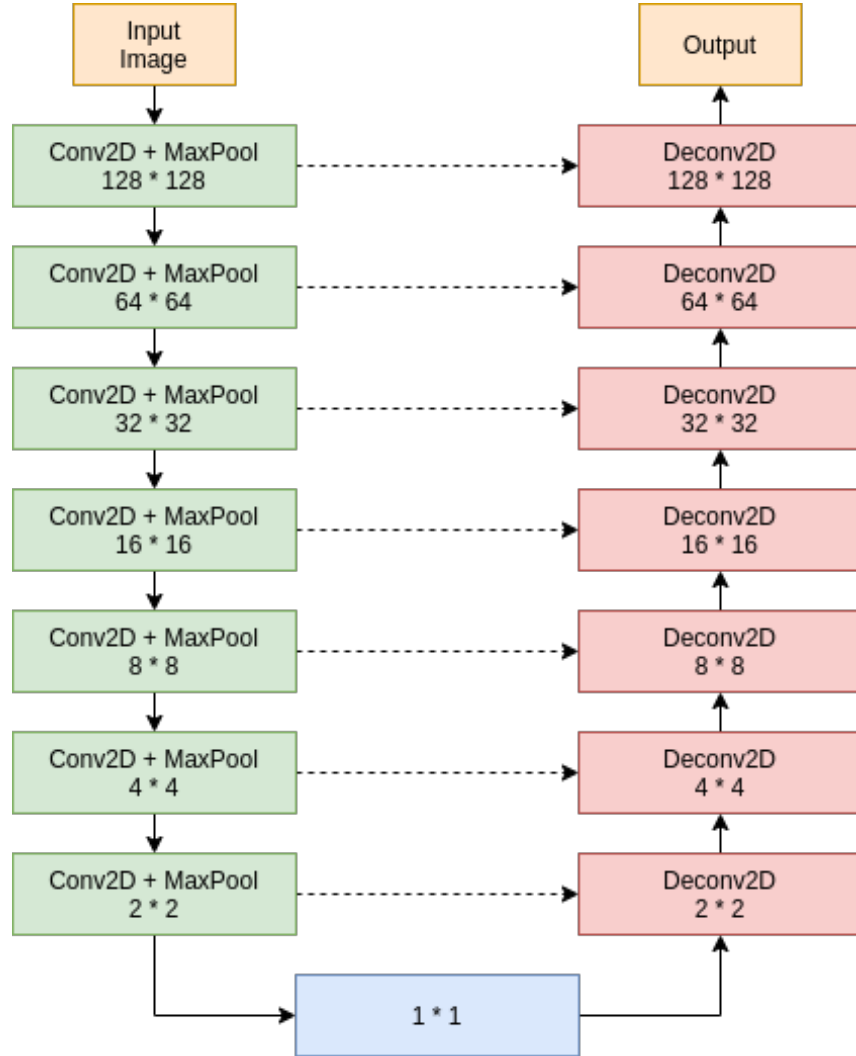


Figure 3.3: Rough Network

The purpose of the rough network is to generate a quick prediction. As our prediction should be a fully restored image, the rough network generates a partially restored image with enough patchiness and blurriness.

3.3.2 Refinement Network

Similar to the rough network, the refinement network also consists of an encoder and a decoder. The input of this network is the output of the rough network conditioned on the original input image (distorted image). The key structural features of this layer are:

- **Encoder:** The encoder has numerous encoder blocks each with one 3×3 convolution and one 4×4 dilated convolution. The 3×3 convolution keeps the spatial size of the image unchanged but doubles the number of channels. The 4×4 dilated convolution reduces the spatial size to half while keeping the number of channels unchanged. So, in each subsequent encoder block in the refinement network, the image is spatially halved and the number of channels is doubled. We have also used a CSA block in the encoder and placed it right after the 3rd encoder block
- **Decoder:** The decoder of the refinement network is symmetrical to the encoder. The decoder consists of numerous decoder blocks instead of encoder blocks and in each block, the 4×4 convolution is replaced with a deconvolution operation. Like rough networks, skip connection from the corresponding encoder layer is also used here.
- **Coherent Semantic Attention:** Though image in-painting is a very difficult task, various deep learning approaches have provided us with excellent results. The problem with most of the models is that, when there is a discontinuity of local pixels in the image, the results tend to have blurry textures and distorted structures. Hongyu Liu Et al. [8] proposed a different approach using a two-step process that involved consistency loss and feature patch discriminator. Their model addressed the general in-painting task just like a human being keeping the semantic relevance and feature continuity in mind. The overall concept of this paper is very close to what we are trying to achieve and helped us a lot while building our own model which aims to achieve better results by using different discriminator models.

The Figure 3.4 shows how the consistency loss works. Each pixel in the blank portion

is dependent on pixels outside the blocked region. During training, the model learns to assign these attention values.

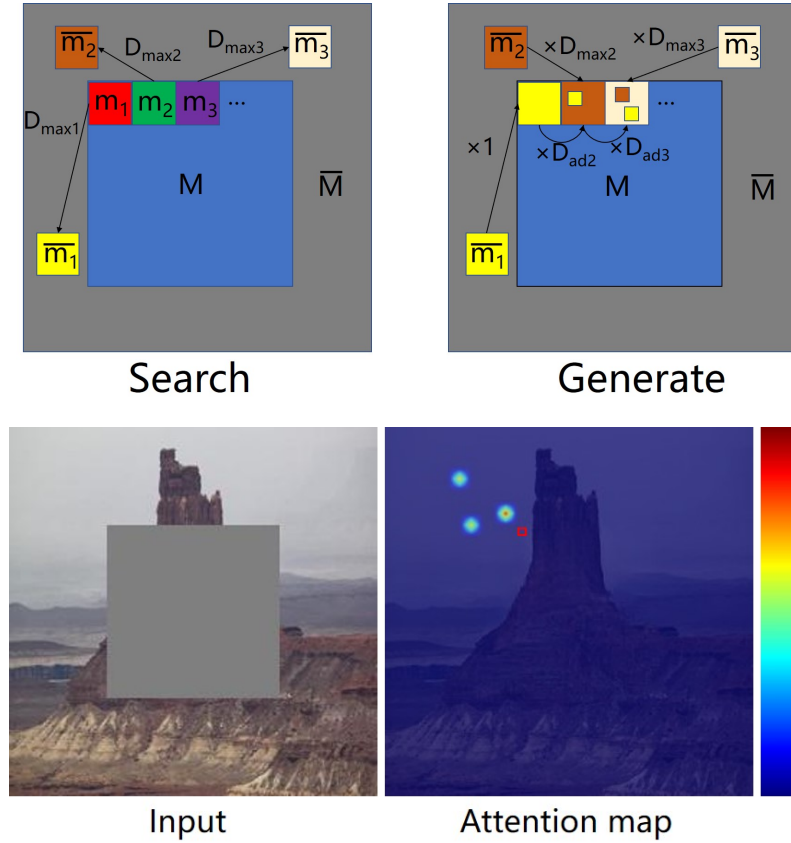


Figure 3.4: Consistency loss (Similar to the work of Liu et al. [8])

- **Loss:** The refinement network primarily uses the adversary loss and the structure mentioned in the section 3.2.

This type of input stacks the information of the known areas to urge the network to capture the valid features faster, which is critical for rebuilding the content of hole regions.

3.4 Discriminator

In this report, along with using two generators, we have used two discriminators as well. Our intuition is that one of the discriminators analyzes global correctness and the other one analyzes local correctness. We have named the first one as Global discriminator and the later one as Patch discriminator.

3.4.1 Patch discriminator

We have developed the patch discriminator in accordance with the patchGAN proposed by Isola et al. [6]. The patch discriminator is trained using the adversarial loss function mentioned in section 3.2. The discriminator is built on the first few layers of pre-trained VGG-16 followed by three 4×4 convolutional blocks.

The Figure 3.5 shows the architecture of patch discriminator. The discriminator is just an image-classifier with several convolutional blocks. The output of the classifier ends few steps prior to the convention of (1,1). In case of the discriminator proposed in this report, the output is 14×14 . Each pixel in this output represents a portion of the original image.

A partially trained model ensures a faster convergence. Since the discriminator and the generator learns simultaneously with the min-max process it is necessary to have a learning balance between them.

3.4.2 Global discriminator

The global discriminator is not too different from the patch discriminator. The only difference is the output shape of the model. Here, the output is directly 1×1 . So, the output gives a global perspective if the overall image is real or not.

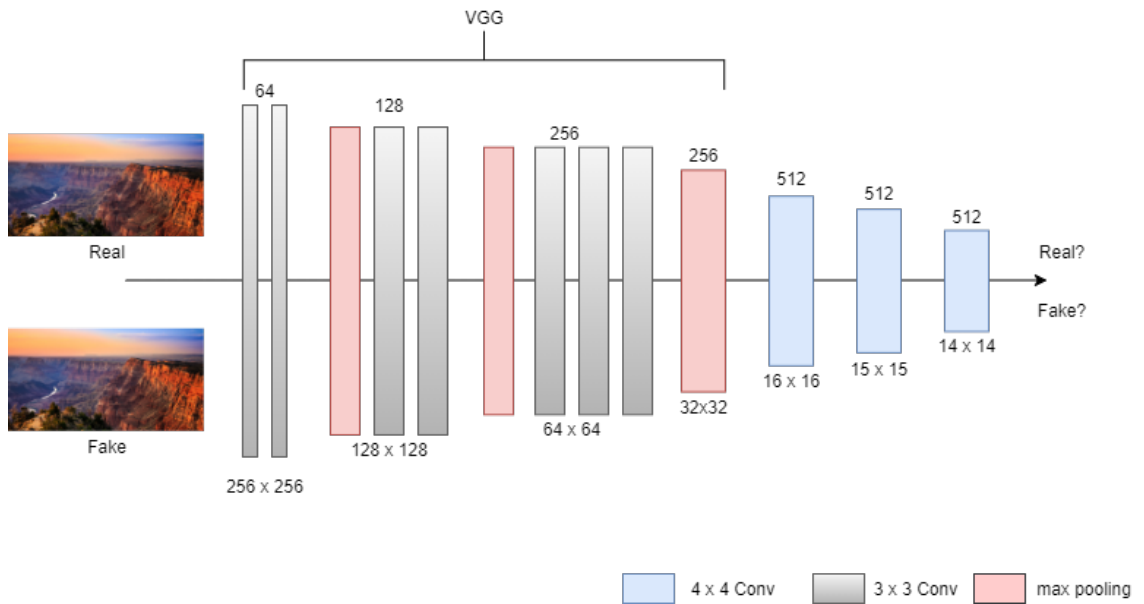


Figure 3.5: Patch discriminator

3.5 Adversarial training

Training a generative model is quite difficult. Conventional method of training does not work. The training objective of such generative models are quite different from the ones used in ordinary CNNs. The training objective used in this report is based on the conditional Generative Adversarial Network.

Chapter 4

Experimental Setup

4.1 Environment

We performed our experiments in Google Colaboratory [41]. Google Colaboratory allows users to execute Jupyter Notebooks and provides access to free GPUs. For a given session, they only offer 1 GPU. The Colab provides 1 12GB Tesla K80 GPU. Tesla K80 is not one of the fastest GPUs in the market, but this is the best we had at our disposal. There is also a time limit on usage. One session can run for 12 hours. A user needs to wait another 12 hours before getting the opportunity to use GPUs for training. Google Colab allocates RAM and CPU configuration according to the session requirements. If the RAM requirement is high, the session automatically shifts to a higher RAM. For our experimentations, the RAM requirement was around 4GB.

We used Google Drive for storage. The Google Colab can be directly mounted on a Drive location. This is the conventional method of running experiments with large datasets in Google Colab. However, there was an issue with this method. As Google Drive is in a remote server, the I/O operation between Colab and Drive is incredibly slow. The workaround for it would be loading the entire dataset to the RAM. But since the size of the dataset is huge, we were not able to opt for this option.

Another viable option for our training was using Kaggle. Kaggle provides Tesla P100 GPU. However, Kaggle does not support easy access to Google Drive storage. There is also a weekly usage limit. Therefore it is more troublesome to train using Kaggle. In Figure 4.1, a performance comparison between Kaggle and Colab has been shown. The training time mentioned in the chart is for the FastAI dataset with batch size 16. [42]

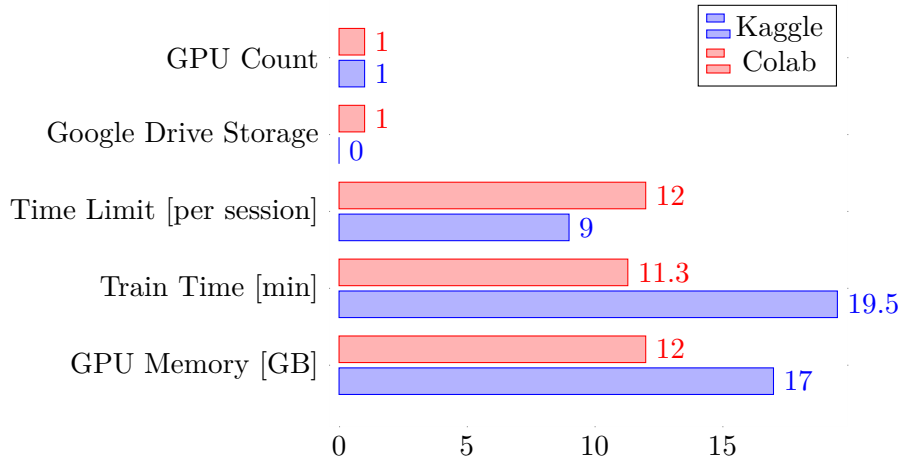


Figure 4.1: Performance comparison between Google Colab and Kaggle

4.2 Hyper-parameters

Due to limited computational resources, the most difficult part of training the model was tuning the hyper-parameters. For external models, we did not tune the hyper-parameters at all. For StructGAN, we tried to find the best options.

Since the generative adversarial network is the most important part of our model, the decision for choosing the GAN type was also vital. After thorough experimentation, we opted for the LSGAN [43].

Another important part of our proposal was the Structural Loss. In the calculation of structural loss, the type of the structural element was vital. We tried experimenting with Laplacian and Sobel structures. The Laplacian structure gave significantly better results in comparison to Sobel structure.

We used initial learning rate $\alpha = 0.002$. For learning rate policy, we used Lambda decay. We used 50 as the decay interval.

The GAN weight and Struct weight were also important determining factor for the performance of the model. We used 20% GAN weight and 20% Struct weight during our experimentations. These values can be further tuned to increase the performance of the model.

4.3 Dataset

We used two distinct datasets for our training. Viz – COCO [44] dataset and Places2 (Val) [45] dataset.

4.3.1 Places 2 (Val)

The Places2 Database [45] was primarily developed for Places365-Challenge, an image-recognition competition. The database consists of 6.2 million extra images in addition to the 1.8 million images from Places365-Standard dataset. Therefore, the total number of train images become 8 million. In our experiments, we only used the validation set of the database to train our models. The validation set contains 36500 images. The database is sub-divided into 365 categories and each category contains 100 images.

4.3.2 COCO

The Common Object in Context (COCO) [44] dataset is a widely used dataset for training deep learning models. The dataset was created by Google to aid future research for object detection, instance segmentation, image captioning, and person key points localization. The motivation of the COCO dataset is to check how well our model understands common objects in context. In the sector of computer vision, the COCO dataset is renowned. It is one of the most popular datasets with 330K images (200K labeled), 1.5 million object

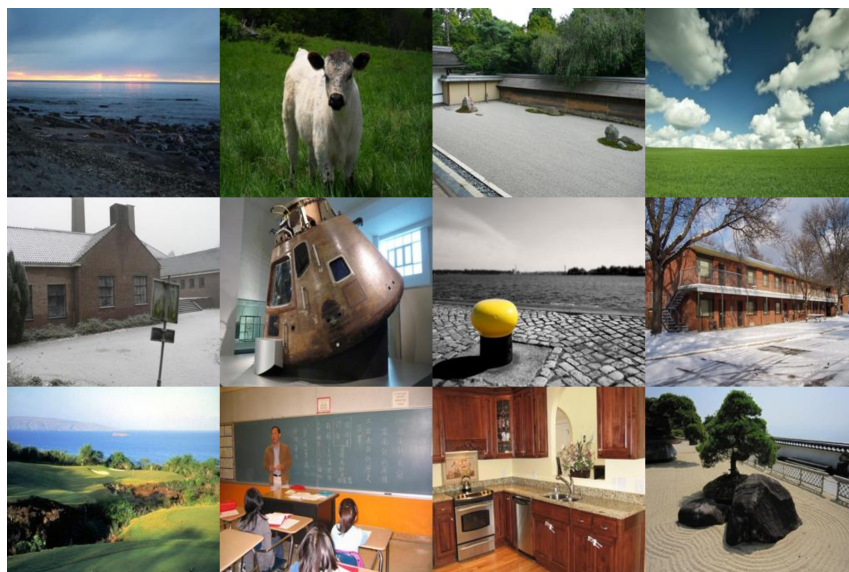


Figure 4.2: Few samples from the Places2 (Val) dataset

instances, 80 object categories, 91 stuff categories, 250,000 people with key points. The biggest advantage of using this dataset is its sheer size. It is one of the best image datasets available, so it is widely used in cutting-edge image recognition artificial intelligence research. It is used in open-source projects such as Facebook Research’s Detectron [46], Matterport’s Mask R-CNN, Endernewton’s Tensorflow Faster RCNN for Object Detection, and others.

The dataset contains segmentation map, object detection annotation and many other feature for each image. Figure 4.3 shows few examples from the COCO dataset with their corresponding segmentation maps.



Figure 4.3: Few samples from the COCO dataset

4.4 Preprocessing

Data preprocessing was essential for our work. We used random transformations to make the model more robust. The input size of the model is $batch_size \times 3 \times 256 \times 256$. The preprocessing step involved converting the data to the appropriate size. For our model, we used `batch_size` to be 1. In most generative adversarial networks, this is the convention. For the normalization layer, we used “instance norm”.

Another important portion of data preprocessing was the addition of the noise. To maintain in-painting convention, we have used a center-square occlusion system. With this system, a fixed-sized square region is cropped from the center of each image and it is replaced with gray color.



Figure 4.4: Example of Center-square Occlusion Mask

Chapter 5

Result and Discussion

5.1 Sample Results

Our model performed fairly well on the Places2 dataset. Figure 5.1, Figure 5.2, and Figure 5.3 are few of the selected samples from our experiments. The outputs shown in Figure 5.1 are mostly natural scenes with less complicated details. Our model performs excellently on this kind of images.

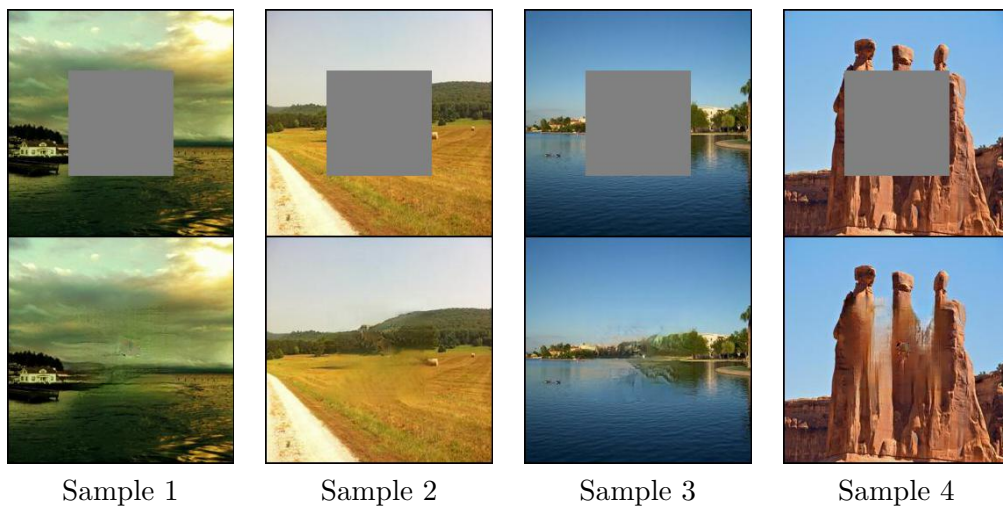


Figure 5.1: Sample Outputs

Images from Figure 5.2 are also natural scenes. However, there are many complicated edges in these images. Our model’s performance on such images is also outstanding. Our model has successfully captured the fine details of these natural scenes.

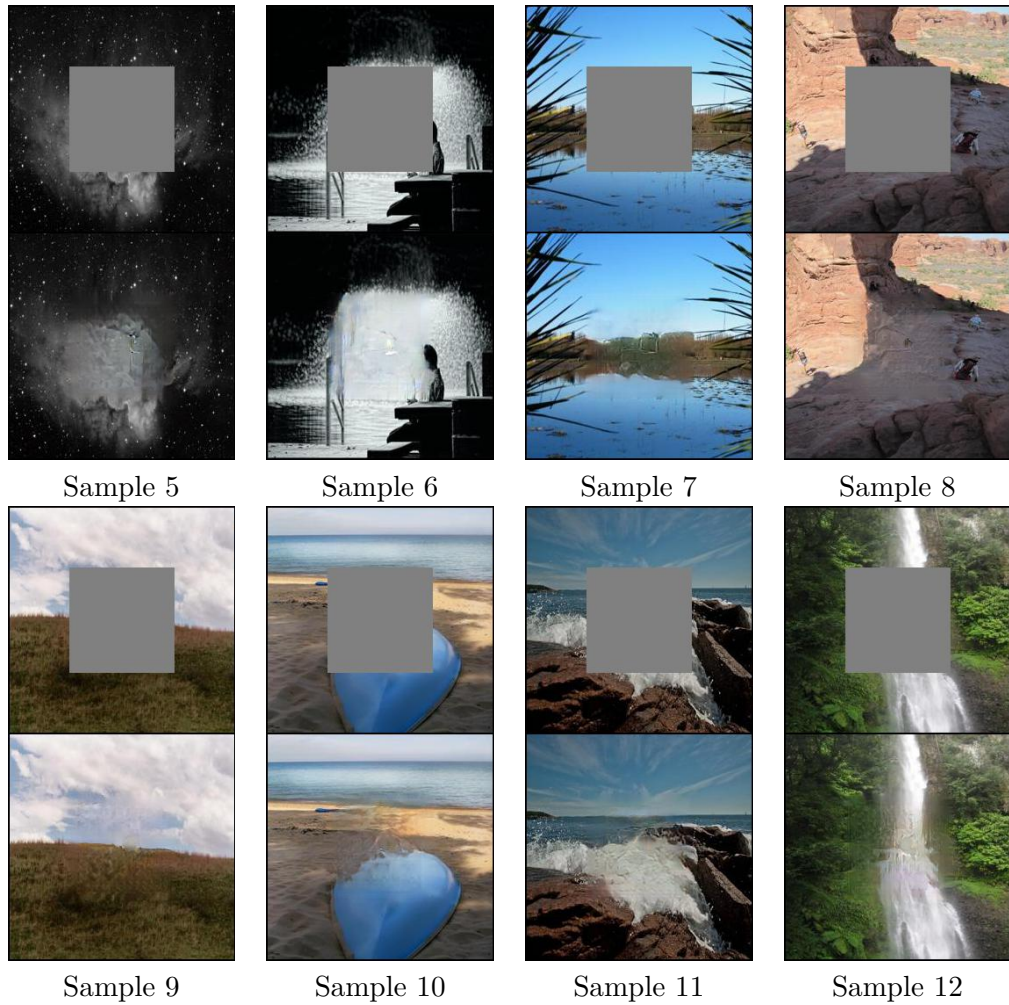


Figure 5.2: Sample Outputs

For images with too non-natural objects, our model does not perform so well. The samples shown in Figure 5.3 are examples of such images. For such objects, our model has produced blurry outputs.

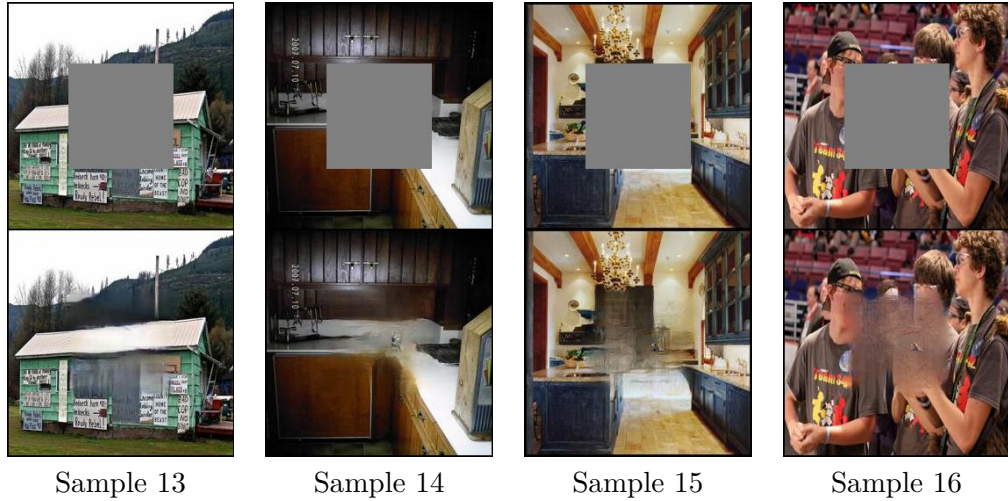


Figure 5.3: Sample Outputs

5.2 Quantitative Analysis

We performed two types of quantitative analyses – Ablation Study and Performance Evaluation over epochs.

5.2.1 Ablation Study

Table 5.1: Quantitative Analysis of our proposed model (StructGAN).

	Gen 1	Gen 2	Disc 1	Disc 2	Struct	SSIM	PSNR	MSE
Pix2pix [6]	UNET_256	-	basic	-	-	0.831	31.72	0.114
CSA [8]	UNET_256	u_csa	basic	feature	-	0.984	69.31	0.079
Struct_Pix2pix	u_struct	-	basic	-	Laplacian	0.852	30.12	0.127
Struct_CSA	UNET_256	u_struct	basic	feature	Sobel	0.91	41.92	0.131
StructGAN	UNET_256	u_struct	basic	feature	Laplacian	0.997	64.77	0.081

The Table 5.1 shows a comparative study of StructGAN with other image in-painting models. The experiments were conducted in our experimental setup. Therefore the scores for the external models can increase if better hyper-parameters are used during training.

The four other relevant models chosen for the ablation study were: Pix2pix [6], CSA [8], Struct_Pix2pix and Struct_CSA. The Struct_Pix2pix was formed by just adding a

Laplacian structural element with the Pix2pix architecture. Similarly, the Struct_CSA was formed by adding a Sobel structural element with the CSA architecture. The models had varying number of generator and discriminator units which was helpful for the ablation study as the effect of having multiple units of generators and/or discriminators could also be sensed.

The model with our proposed novelty produced the best SSIM value of 0.997 in comparison with the other models (0.984, 0.91, 0.852 and 0.831). This indicated that the structural consistency is being maintained in our produced images as the Structural Similarity Index Metric (SSIM) is very good at finding the structural similarity between two images. Our model is slightly lagging behind when it comes to the PSNR and MSE values compared to the other ones. We hope that these values will get better with more training. However, getting the highest SSIM value was a huge success for us as this metric independently focuses on the structural consistency between two images, which was our main motivation behind this research.

5.2.2 Performance Evaluation

Judging the performance of the model from the produced images can be highly subjective. We did a bit of background study about the metrics that can be used for this purpose. After a careful selection, we selected three widely used quantitative metrics for our evaluation. The metrics were Structural Similarity Index Metric (SSIM), Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE). We wrote independent scripts using Python to find the values of these metrics between images.

The core metric that we have worked on is the Similarity Index Metric (SSIM). The reason behind this was, this metric is very good at finding the structural similarities between two images. The higher the SSIM value, the more the structural consistency between two images. Our model has produced a value of 0.997 which is better than all the other

models.

Figure 5.4 shows a Number of steps vs SSIM value graph. As we can see from the graph, the SSIM value kept on increasing as the number of steps increased, This shows that our model got better and better in producing more structurally consistent images over time.

The Peak Signal-to-Noise Ration (PSNR) is a metric which is used to measure the level

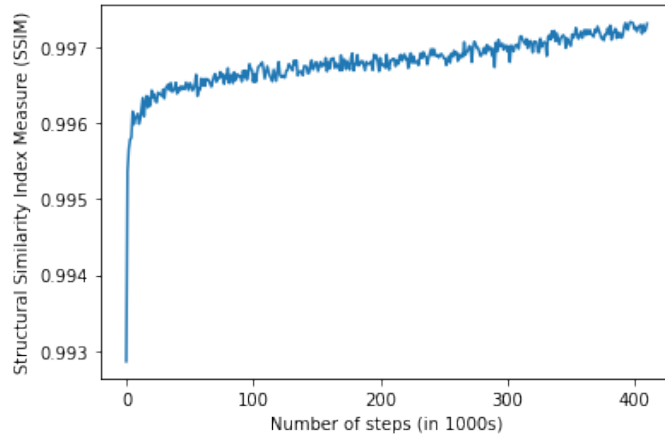


Figure 5.4: SSIM Score of StructGAN

of noise removed from the output image when compared to the input. It is an excellent metric when it comes to Noise Removal tasks. Our model produced a slightly less PSNR value (64.77) in comparison with the CSA model. We hope that this value will get better with more time and some fine tuning.

Figure 5.5 shows a Number of steps vs PSNR value graph. As we can see from the graph, the PSNR value increased over time which shows the model got better in producing images with less noise compared to the input.

The third and final metric is one of the most common metrics used in the domain of Computer Vision, Mean Squared Error (MSE). It takes the average of squared differences among the pixel values of the input and output images. Our model has produced a MSE value of 0.081 which will decrease as we train it more.

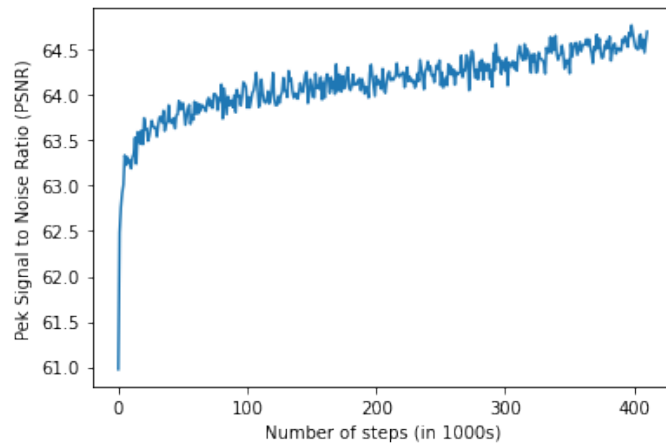


Figure 5.5: PSNR Score of StructGAN

Figure 5.6 shows a Number of steps vs MSE value graph. As we can see from the graph, the MSE value decreased over time which shows the model got better in producing images with less mean squared error which was expected.

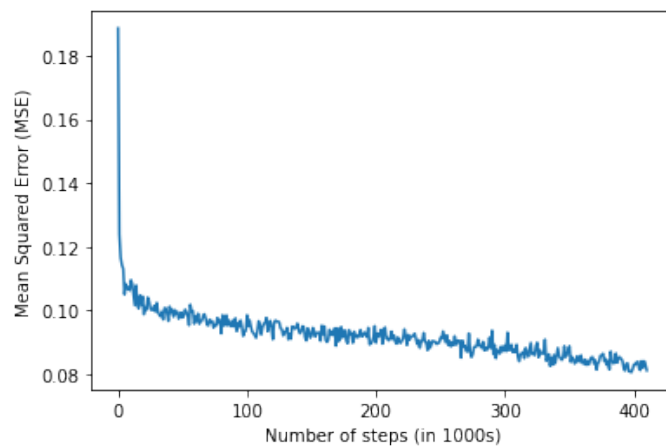


Figure 5.6: MSE Score of StructGAN

5.3 Qualitative Analysis

Image reconstruction is not a yes-no task. Therefore, off-the-shelf metrics like SSIM, PSNR, etc. are not able to capture the overall quality of the model. For this reason, it is essential to perform qualitative analysis as well.

The sample outputs from Figure 5.1, Figure 5.2 are excellent and most of the images are almost perfect. These are the general conditions of our generated images that include scenic views. So, we can say, the model generates great results for natural scenes that have low detail.

The samples from Figure 5.3 are not as good as the previous ones. The original images of these samples had much more details. Our model failed to generate such details. The reason for this failure is the lack of information in the surrounding. The texts, or the human face structure that went missing in these generated outputs are very complex. To have a model that can generate outputs with such details, we need to train the model on one particular object. For face generation, we could train the model on faces dataset.

5.4 Discussion

In light of Section 5.2 and Section 5.2, we can say that our proposed model performs fairly well. Considering the complexity of the task and despite the computational limitation during the training period, the model has learned to generate images of excellent quality that look almost real to normal eyes.

Our intuition behind structure loss was that the model will learn to understand image structure of a region from its surrounding regions. Our model has a very high SSIM score which indicates to the fact that our model is successfully understanding the structural consistency. However, our model under-performs in case of PSNR. The reason for it is, our model is not able to generate sharp edges.

Chapter 6

Conclusion

6.1 Summary

Our research aimed at maintaining the structural consistency of an image while restoration using a novel approach. The existing methods of image inpainting created blurry textures while filling in the missing pixel values. Our model was motivated from the cognitive behavior of a human being while restoring the image.

For maintaining the structural consistency, we proposed a novel loss function, which uses a Laplacian structural element along with two other losses (Consistency Loss and Refinement Loss), to achieve its goal. The overall architecture consists of two generator networks (Rough Network and Refinement Network) and two discriminator networks (Patch Discriminator and Structure Discriminator).

Judging the efficiency of the model from just the qualitative results can be a cumbersome. The quality of inpainted output images will highly vary from person to person. Finding a good quantitative metric to perfectly judge the performance of our model was difficult. Studying the previous researches in this sub-domain, we have selected three metrics to perform the quantitative analysis; Structural Similarity Index Metric (SSIM), Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE). We have performed an ablation

study including a total of 5 relevant models. So far, the model with our proposed novelty and architecture has outperformed the other models. It could produce a SSIM value of 0.997 which is the best SSIM value when compared to the other models. However, the model could not perform as good as the CSA model in terms of the PSNR and MSE values. We really hope that the values will keep on getting better as we keep on training the model.

Given the complexity of our model, finding sufficient computational resources was a constant challenge for us. We have used Google Colaboratory for the whole implementation. We are still training our model and the results are becoming satisfactory with time. Since the main goal of our research was to produce synthetic images keeping the structural consistency in focus, we can say that, with the current SSIM value of 0.997, the research goal has been successfully reached.

6.2 Future Works

We have developed a custom script using Python that can produce realistic noisy images. The script incorporates various kinds of noise. The script randomly chooses one or more of the following types of noise to prepare the image.

- *Doodle noise*: A doodle is a rough line drawn with a brush which generally replaces the original pixels with a solid color. This noise is important for our research because in many cases, old photos have similar noises where a certain portion of the image is destroyed because of folding. Doodle creates a similar effect and helps our model to learn how to generate the pixels lost due to the doodle noise.

While running, the algorithm mainly selects three things randomly: the brush size, the doodle length and the doodle color. More attributes (the starting point, the ending point, whether the doodles will be connected or not etc.) are flexible and can be changed as needed.

- *Salt and pepper noise*

- *Gaussian noise*
- *Poisson noise* [47]

The algorithm assigns different priorities to each noise. We have not incorporated this noise script into our training phase due to the lack of a global benchmark. However, we have tried to show output of our model for these types of noises. The output was underwhelming because the model was not trained on such type of noise. In future, we would like to develop a new benchmark for our doodle noise. We would then train the model to maximize the accuracy for the benchmark.

We would also like to develop a custom metric that best describes correctness for a realistic broken image. This new metric can be designed based on coherent correctness of the image itself. This metric needs a lot of further study.

References

- [1] R. Dahl, M. Norouzi, and J. Shlens, “Pixel recursive super resolution,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5439–5448.
- [2] X. Li, G. Hu, J. Zhu, W. Zuo, M. Wang, and L. Zhang, “Learning symmetry consistent deep cnns for face completion,” *IEEE Transactions on Image Processing*, vol. 29, pp. 7641–7655, 2020.
- [3] “Wikipedia: Total variation denoising,” https://en.wikipedia.org/wiki/Total_variation_denoising, accessed: 2020-09-28.
- [4] “Before and after comparisons of adobe’s amazing image deblurring feature,” <https://www.slrlounge.com/zoom-and-enhance-google-brain-super-resolution-tech-make-tv-trope-a-reality/>, accessed: 2020-09-28.
- [5] “Learning generative adversarial networks (gans),” <https://laptrinhx.com/learning-generative-adversarial-networks-gans-2910834212/>, accessed: 2021-03-01.
- [6] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

- [7] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, “Generative image inpainting with contextual attention,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5505–5514.
- [8] H. Liu, B. Jiang, Y. Xiao, and C. Yang, “Coherent semantic attention for image inpainting,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 4170–4179.
- [9] J. G. Schavemaker, M. J. Reinders, J. J. Gerbrands, and E. Backer, “Image sharpening by morphological filtering,” *Pattern Recognition*, vol. 33, no. 6, pp. 997–1012, 2000.
- [10] A. Polesel, G. Ramponi, and V. J. Mathews, “Image enhancement via adaptive unsharp masking,” *IEEE transactions on image processing*, vol. 9, no. 3, pp. 505–510, 2000.
- [11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.
- [12] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.
- [13] J. Zhang, D. Zhao, R. Xiong, S. Ma, and W. Gao, “Image restoration using joint statistical modeling in a space-transform domain,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 6, pp. 915–928, 2014.
- [14] Y. Tai, J. Yang, X. Liu, and C. Xu, “Memnet: A persistent memory network for image restoration,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 4539–4547.
- [15] M. K. Mihcak, I. Kozintsev, K. Ramchandran, and P. Moulin, “Low-complexity image denoising based on statistical modeling of wavelet coefficients,” *IEEE Signal Processing Letters*, vol. 6, no. 12, pp. 300–303, 1999.

- [16] M. K. Mihcak, I. Kozintsev, and K. Ramchandran, "Spatially adaptive statistical modeling of wavelet image coefficients and its application to denoising," in *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No. 99CH36258)*, vol. 6. IEEE, 1999, pp. 3253–3256.
- [17] G. Fan and X.-G. Xia, "Wavelet-based statistical image processing using hidden markov tree model," in *Proc. 34th Annual Conference on Information Sciences and Systems, Princeton, NJ, USA*, 2000.
- [18] A. Pizurica, W. Philips, I. Lemahieu, and M. Acheroy, "A joint inter-and intrascale statistical model for bayesian wavelet based image denoising," *IEEE Transactions on Image Processing*, vol. 11, no. 5, pp. 545–557, 2002.
- [19] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2. IEEE, 2005, pp. 60–65.
- [20] S. Lefkimmiatis, "Non-local color image denoising with convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3587–3596.
- [21] C.-A. Deledalle, J. Salmon, A. S. Dalalyan *et al.*, "Image denoising with patch based pca: local versus global." in *BMVC*, vol. 81, 2011, pp. 425–455.
- [22] J. Wang, Y. Guo, Y. Ying, Y. Liu, and Q. Peng, "Fast non-local algorithm for image denoising," in *2006 International Conference on Image Processing*. IEEE, 2006, pp. 1429–1432.
- [23] T. Chen, K.-K. Ma, and L.-H. Chen, "Tri-state median filter for image denoising," *IEEE Transactions on Image processing*, vol. 8, no. 12, pp. 1834–1838, 1999.
- [24] M. Zhang and B. K. Gunturk, "Multiresolution bilateral filtering for image denoising," *IEEE Transactions on image processing*, vol. 17, no. 12, pp. 2324–2333, 2008.

- [25] S. M. LoPresto, K. Ramchandran, and M. T. Orchard, “Image coding based on mixture modeling of wavelet coefficients and a fast estimation-quantization framework,” in *Proceedings DCC’97. Data Compression Conference*. IEEE, 1997, pp. 221–230.
- [26] N. R. Goodman, “Statistical analysis based on a certain multivariate complex gaussian distribution (an introduction),” *The Annals of mathematical statistics*, vol. 34, no. 1, pp. 152–177, 1963.
- [27] S. Cho and S. Lee, “Fast motion deblurring,” in *ACM SIGGRAPH Asia 2009 papers*, 2009, pp. 1–8.
- [28] L. Xu and J. Jia, “Two-phase kernel estimation for robust motion deblurring,” in *European conference on computer vision*. Springer, 2010, pp. 157–170.
- [29] S. K. Nayar and M. Ben-Ezra, “Motion-based motion deblurring,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 6, pp. 689–698, 2004.
- [30] J. Biemond, R. L. Lagendijk, and R. M. Mersereau, “Iterative methods for image deblurring,” *Proceedings of the IEEE*, vol. 78, no. 5, pp. 856–883, 1990.
- [31] P. C. Hansen, J. G. Nagy, and D. P. O’leary, *Deblurring images: matrices, spectra, and filtering*. SIAM, 2006.
- [32] Q. Shan, J. Jia, and A. Agarwala, “High-quality motion deblurring from a single image,” *Acm transactions on graphics (tog)*, vol. 27, no. 3, pp. 1–10, 2008.
- [33] B. Sahu, “Towards data science: An evolution in single image super resolution using deep learning,” <https://towardsdatascience.com/an-evolution-in-single-image-super-resolution-using-deep-learning-66f0adfb2d6b>, 2019, accessed: 2020-09-29.
- [34] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

- [35] C. Dong, C. C. Loy, K. He, and X. Tang, “Image super-resolution using deep convolutional networks,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2015.
- [36] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, “Photo-realistic single image super-resolution using a generative adversarial network,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4681–4690.
- [37] J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution,” in *European conference on computer vision*. Springer, 2016, pp. 694–711.
- [38] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy, “Esrgan: Enhanced super-resolution generative adversarial networks,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 0–0.
- [39] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, “Loss functions for image restoration with neural networks,” *IEEE Transactions on computational imaging*, vol. 3, no. 1, pp. 47–57, 2016.
- [40] X. Wang, “Laplacian operator-based edge detectors,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 5, pp. 886–890, 2007.
- [41] “Google colab,” <https://colab.research.google.com/>, accessed: 2010-09-30.
- [42] “Kaggle vs. colab faceoff — which free gpu provider is tops?” <https://towardsdatascience.com/kaggle-vs-colab-faceoff-which-free-gpu-provider-is-tops-d4f0cd625029>, accessed: 2020-01-02.
- [43] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, “Least squares generative adversarial networks,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2794–2802.

- [44] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *European conference on computer vision*. Springer, 2014, pp. 740–755.
- [45] B. Zhou, A. Khosla, A. Lapedriza, A. Torralba, and A. Oliva, “Places: An image database for deep scene understanding,” *arXiv preprint arXiv:1610.02055*, 2016.
- [46] A. Joulin and F. Paris, “Facebook ai research,” *Learning Visual Features from Large Weakly Supervised Data*, 2015.
- [47] “Poisson noise - wikipedia,” https://en.wikipedia.org/wiki/Shot_noise, accessed: 2010-09-30.