De-raining Natural Scenes using Diffusion Models

Emil Biju
Electrical Engineering
Stanford University
emilbiju@stanford.edu

Sidharth Tadeparti
Mechanical Engineering
Stanford University
sidt@stanford.edu

Sneha Jayaganthan
Electrical Engineering
Stanford University
snehajay@stanford.edu

Abstract

Rain significantly degrades the quality of images captured in outdoor environments, posing challenges for various computer vision applications. In this project, we explore multiple advanced techniques for rain removal from natural scenes, with a strong focus on the generative and denoising power of diffusion models. Our initial approach combines the segmentation capabilities of a fine-tuned U-Net with conditional diffusion models. By fine-tuning the U-Net to accurately segment rain-affected regions, we apply the conditional diffusion model exclusively to these localized areas to remove rain artifacts while preserving the underlying scene details.

We also explore end-to-end image translation for deraining using a fine-tuned diffusion model and a GAN-based Pix2Pix model. We further experiment with data-efficient approaches using style transfer techniques based on both VGG and CycleGAN architectures. For training and evaluation, we utilize datasets provided in the WeatherDiffusion repository repository and synthetic dataset for low-cost model training. To compare the performance of each technique, we benchmark them using the PSNR and SSIM metrics. The results are promising, with the best result obtained using a fine-tuned diffusion model which generates de-rained images that closely match the ground truth images in terms of visual quality. Our comprehensive comparative analysis highlights the tradeoffs in terms of computational complexity, data efficiency, and output quality.

1. Introduction

Rain is a common weather phenomenon that significantly affects the quality of images captured in outdoor environments. This degradation can pose substantial challenges in various computer vision applications such as autonomous driving, surveillance, and outdoor scene analysis. In recent times, Diffusion models have gained success in a wide variety of image generation tasks such as image editing, denoising, and conditional generation. In this project, we study the efficacy of diffusion models for de-raining images in addition to adjacent generative modeling techniques such as GANs. We carry out this study systematically to understand performance trade-offs while sequentially using models of increasing computation intensity.

In this project, there are three distinct approaches that we use. The first is the use of stable diffusion's [9] in-painting capability to form a baseline. The second is the use of style transfer, and the third is the use of a custom pre-trained diffusion model. A U-Net-based segmentation model is also employed in an attempt to accelerate diffusion inference. This segmentation model is also used to generate the baseline stable diffusion outputs.

The inputs are the datasets containing scenes that have rain and the corresponding ground truth scenes under non-rainy conditions. The expected output is a de-rained version (without rain drops) of the rainy image. It is to be noted that it is often quite hard to collect real-world data of such pairs and therefore, we present a discussion on using a synthetic dataset. We also examine the efficacy of style transfer techniques of different types, to understand if it can be used to drive data efficiency fpr. The models will be built and primarily evaluated on the PSNR (Peak Signal-to-Noise Ratio)[5] and SSIM (Structural Similarity Index)[10] metrics apart from analyzing the overall visual quality.

This approach for de-raining using diffusion models finds high utility for diverse applications, such as in improving the road visibility for autonomous vehicles, improving the clarity of images taken under rainy conditions, or even creating datasets for training weather simulation models.

In conclusion, the technical contributions of this work are the following:

- Benchmarking of various generative approaches for deraining applications.
- Develop techniques to accelerate inference of diffusion-based de-raining to improve compute efficiency.
- Examine methods to enable data efficiency while deraining.

2. Related Work

Early approaches for rain simulation in images focused on physical models to simulate rain streaks or utilized traditional image processing techniques. These methods often utilized temporal and spatial information to identify and remove rain streaks. For instance, [2] Garg and Nayar proposed a method that models rain streaks as noise and employs spatio-temporal smoothing to remove them from video sequences. While we believe in their effectiveness for certain scenarios, these methods struggle with complex backgrounds and varying rain intensities, often leading to artifacts and loss of image details. With the advent of machine learning, more sophisticated approaches were introduced. One notable method is the work by [7] Kang et al., which utilized sparse coding to separate rain streaks from the background. This approach improved over traditional methods by learning a dictionary of rain streaks, allowing for better rain removal. However, it still faced limitations in handling diverse rain patterns and intensities.

The introduction of deep learning revolutionized the field, leading to significant advancements in rain removal techniques. [1] Fu et al. proposed a deep convolutional neural network (CNN) to directly learn the mapping from rainy images to clean images. Their model, called DerainNet, demonstrated superior performance over the traditional methods. However, CNNs often struggle with generating high-quality, realistic images as evident with the observed outputs that weren't smooth. This limitation arises because CNNs are primarily discriminative models that excel in feature extraction but lack the sophisticated generative capabilities. Another particularly clever approach was proposed by [8] Li et al., who introduced a recurrent neural network (RNN) based model that iteratively removes rain streaks. This model was able to handle varying rain densities and patterns by progressively refining the output image over a number of timesteps. However, RNNs can be computationally expensive and slow during both training and inference and often struggle with modeling long-term dependencies. This makes them less suitable for real-time applications where quick processing is required. Recently, conditional generative adversarial networks (CGANs) have been introduced for De-raining images. For instance, [12] and [13] proposed a CGAN-based method for De-raining which incorporates architectural novelties in the generator-discriminator pair as well as constraints on the loss function for optimizing the visual quality. We believe these methods are state-of-the art and the above works are directly relevant to our project as they highlight the effectiveness of adversarial training in image restoration as well as provide a foundational understanding of how to incorporate complex loss functions to achieve high-quality outputs. Moreover, Diffusion models have also gained popularity for image restoration tasks in recent times, and these operate by iteratively diffusing information across image pixels to remove noise or artifacts. [**?**] Section et al. proposed a patch-based restoration algorithm that employs a conditional denoising process on diffusion models, specifically focusing on localized patches to address weather-induced distortions, which aligns closely with our approach of targeted rain removal. [4]Ho et al. also utilized a stochastic process to iteratively refine the image, synthesizing high quality images using diffusion probabilistic models. These works are particularly clever in their approach and are directly relevant to our project as they demonstrate both the potential and the requisite methodology of utilizing diffusion models for restoring images affected by specific weather conditions, while providing a basis for our targeted de-raining strategy.

[3] Gatys et al., introduces the concept of using deep convolutional neural networks for style transfer. The methodology involves separating and recombining the content and style of images, which can be applied to tasks such as rain removal by treating the non-rainy image as the style. Style-transfer techniques such as [6] have also utilized conditional adversarial networks for image-to image translation where the authors have developed the Pix2Pix framework, that can be adapted for various image translation tasks, including weather condition changes by learning the mapping between paired datasets of different styles. These works are relevant to our project as similar style-transfer techniques can be adapted to our use-case by transferring style from sunny to rainy weather conditions, highlighting the potential for enhancing realism of de-rained images by leveraging domain-specific knowledge.

In our project, we aim to use Conditional Diffusion Models to address the problem of removing rain artifacts and to enhance the image clarity. Moreover, in order to optimize the computational efficiency and ensure faster training, we fine-tune a Pre-trained U-Net model to identify and segment regions in the image affected by rain. With the segmented rain-affected regions identified, we apply the output of the diffusion models to only these localized areas to remove the rain artifacts, while preserving the rest of the image regions. We compare this approach to a baseline method, where we use Stable Diffusion's Inpainting to fill in the regions (marked by the masks obtained from U-Net) using surrounding pixel values. We also compare our de-rained output to the output obtained from using only the diffusion model (without U-Net).

Subsequently, we explore alternate approaches using GANs and style transfer. We also attempt to further fine-tune the diffusion model in order to improve performance. Each of these approaches presents a different set of trade-offs in terms of computational intensity, visual quality, and scalability.

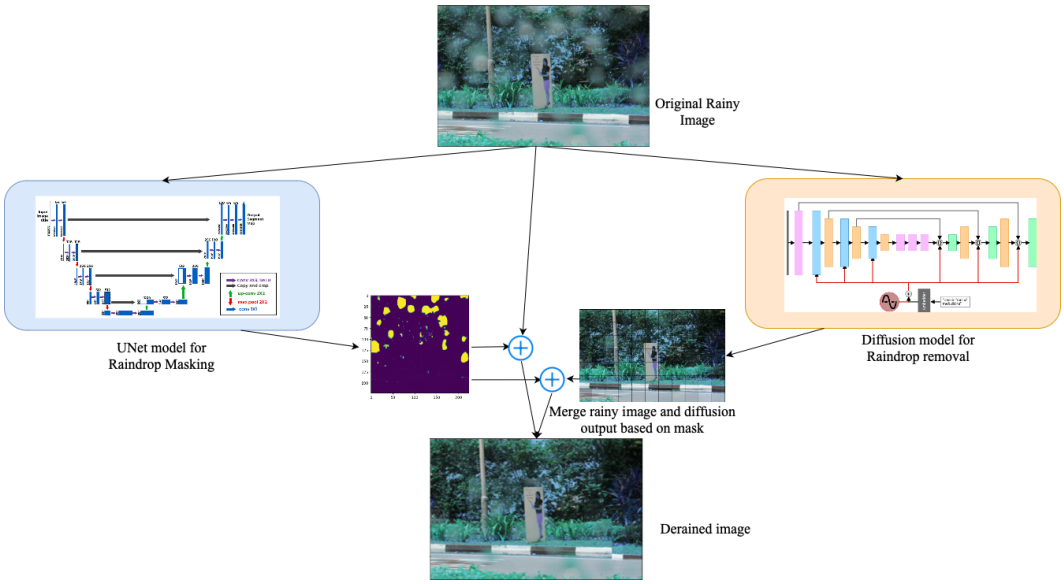

Figure 1. Flowchart indicating the methodology of our De-Raining Model: Diffusion Model + Pre-Trained U-Net

3. Methodology

3.1. Baseline Method: Inpainting

To establish a baseline method, we consider stable diffusion-based inpainting where regions having raindrops are first identified, masked out, and then filled using inpainting based on information from the surrounding pixels. To generate masks corresponding to regions having raindrops, we use a UNet model that we further describe in Section 3.3. The text prompt that is used to guide the model is "Fill in with the environment without rain". However, this baseline method might struggle with highly complex or textured regions, where surrounding pixel values alone become insufficient for realistic inpainting.

To tackle this challenge, we explore end-to-end models trained explicitly on a dataset of rainy and derained images in the next section. We further consider ways to improve computational efficiency and reduce demand for paired rainy and derained images which are difficult to obtain.

3.2. End-to-end models with paired, real-world data

1. Pix2Pix GAN: Pix2Pix uses a conditional GAN to perform image translation and has been successful at a wide variety of tasks such as translating sketches to natural-looking images, satellite images to street maps, grayscale images to RGB images, etc. Hence, we explore the use of Pix2Pix for the deraining task. Given a reference image x and a noise vector z, the GAN generates an image $G(x, z)$ that is expected to resemble the ground truth y which is a translated version of x. Pix2Pix uses a combination of two loss functions. The GAN minimax loss ($\mathcal{L}_{cGAN}$) jointly trains the generator and discriminator with the discriminator conditioned on the reference image x to encourage the generated images to align with the structure of the reference image:

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

It also uses an L1 loss ($\mathcal{L}_{L1}$) that encourages closeness between the generated image and ground truth:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|] \quad (2)$$

The overall objective is given by:

$$G = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G) \quad (3)$$

We adapt this framework to train Pix2Pix on 200 pairs of rainy and derained images of the same scene where x is the rainy image and y is the derained image.

2. Patch-wise Conditional Diffusion Model We use a conditional diffusion model that takes a rainy image as input and predicts a de-rained version of the image. Diffusion models iteratively transform a noisy input into a clean output through a series of denoising steps, effectively removing noise and artifacts. Our initial approach is to train the model on pairs of rainy and de-rained images of the same scene. The forward process of the diffusion model will be a fixed Markov chain that iteratively adds Gaussian noise to the rainy image. The reverse process will iteratively denoise the image by predicting the mean, μ_θ, of the Gaussian transitions conditioned on the original rainy image, given

by:

$$p_\theta\left(\mathbf{x}_{0:T}\right)=p\left(\mathbf{x}_T\right)\prod_{t=1}^{T}p_\theta\left(\mathbf{x}_{t-1}\mid\mathbf{x}_t\right),$$
$$p_\theta\left(\mathbf{x}_{t-1}\mid\mathbf{x}_t\right)=\mathcal{N}\left(\mathbf{x}_{t-1};\boldsymbol{\mu}_\theta\left(\mathbf{x}_t,t\right),\boldsymbol{\Sigma}_\theta\left(\mathbf{x}_t,t\right)\right) \tag{4}$$

The last stage of the denoising process will generate the de-rained version ($\mathbf{x}_0$). The WeatherDiffusion repository implements conditional diffusion, and we have adapted it to our use-case. Specifically, when using conditional diffusion, we condition the reverse process on the reference image x, which in this use case is the rainy image.

$$p_\theta\left(\mathbf{x}_{0:T}\mid x\right)=p\left(\mathbf{x}_T\right)\prod_{t=1}^{T}p_\theta\left(\mathbf{x}_{t-1}\mid\mathbf{x}_t,x\right) \tag{5}$$

The reverse probability distribution is modeled using the noise estimator network $\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t,x,t\right)$, which has an encoder-decoder structure. The reverse sampling process can be carried out as follows,

$$\mathbf{x}_{t-1}=\sqrt{\bar{\alpha}_{t-1}}\left(\frac{\mathbf{x}_t-\sqrt{1-\bar{\alpha}_t}\cdot\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t,x,t\right)}{\sqrt{\bar{\alpha}_t}}\right)+\sqrt{1-\bar{\alpha}_{t-1}}\cdot\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t,x,t\right)$$

where α is a cumulative parameter. The authors of [**?**] provide a diffusion model that is pretrained on multiple weather removal tasks, including de-raining, de-hazing, and de-snowing. The pretrained model serves as an excellent starting point. However, we also fine-tune the model on our paired deraining dataset and analyze the improvement in results. This fine-tuning is performed for 100 epochs on a H100 GPU. A pertinent challenge with the large diffusion model is the high computation complexity and storage space required for deployment on edge devices.

3.3. Improving compute efficiency with rain patch detection

The diffusion model discussed above poses two key challenges. Firstly, it splits the image into patches and runs each patch separately through the diffusion model. This leads to each patch having slightly different contrasts, making the overall image look unnatural, as seen in Figure 3 (middle). Secondly, running every patch through the diffusion model is computationally inefficient as some patches may not have raindrops in them and therefore do not need to be derained.

To solve this, we develop a UNet-based binary segmentation model that predicts regions having raindrops in the image. U-Net is a convolutional neural network architecture designed for image segmentation tasks. It consists of an encoder-decoder structure with skip connections that preserve spatial information across layers. The encoder captures contextual information by downsampling the input image, while the decoder reconstructs the segmentation map by upsampling. To prepare training data for the UNet model, we take each rainy image x and its corresponding ground truth image y from our paired dataset. The mask is then obtained as $\text{Mask}(x,y)=\mathbf{1}(x\neq y)$. This means that the mask takes on a value of 1 where the rainy and ground truth images show a mismatch indicating the presence of raindrops, and 0 at other pixels.

We then take a pretrained U-Net model and fine-tuned it on the dataset to detect and segment rain-affected regions in images. After exploring various hyperparameter and architecture options, we obtained the best results when using resnet50 as the backbone network and AdamW as the optimizer with learning rate of 1e-3. The Dice Loss function is used for training. Once the binary segmentation masks are obtained, as shown in Figure 1, the patches in the original rainy image which are found to contain raindrops are passed through the diffusion model. The corresponding derained outputs are used to replace these patches in the rainy image while the remaining patches stay intact. This step minimizes the patch artifacts introduced from using all patch-wise outputs from the diffusion model and preserves information from the original image. Moreover, the UNet and diffusion model can be run independently and in parallel, thus reducing latency.

3.4. Improving data efficiency with Style Transfer & Synthetic Data

3.4.1 Style Transfer using VGG:

Our approach leverages the neural style transfer algorithm introduced by [3] Gatys et al., which uses convolutional neural networks to transfer the style of one image onto the content of another. The algorithm can be summarized in the following steps:
Feature Extraction: We use a pre-trained VGG19 network to extract high-level features from both the content (rainy) and style (non-rainy) images.
Gram Matrix Calculation: We compute the Gram matrices for the style features to capture the correlations between different filter responses.
Loss Functions: We define the content and style loss functions to measure the differences between the target image and the content/style images.
Optimization: We use gradient descent to iteratively update the target image to minimize the total content/style loss.We represent the rainy image as I_c, the non-rainy style image as I_s, and the target derained image as I_t.

Feature Extraction: Let $F_l(x)$ be the feature maps of layer l in the VGG19 network for image x. For the content image I_c, we denote its feature maps as $F_l(I_c)$. For the style image I_s, we denote its feature maps as $F_l(I_s)$.

Gram Matrix: The Gram matrix $G_l(x)$ for the fea-

ture maps $F_l(x)$ is defined as:

$$G_l(x) = F_l(x)F_l(x)^T$$

Loss Functions
Content Loss: Measures the difference between the target image I_t and the content image I_c at a specific layer l:

$$\mathcal{L}_{\text{content}} = \frac{1}{2}\sum_{i,j}(F_l(I_t)_{ij} - F_l(I_c)_{ij})^2$$

Style Loss: Measures the difference between the Gram matrices of the target image I_t and the style image I_s across multiple layers:

$$\mathcal{L}_{\text{style}} = \sum_l w_l \cdot \frac{1}{4N_l^2 M_l^2}\sum_{i,j}(G_l(I_t)_{ij} - G_l(I_s)_{ij})^2$$

Here, w_l is the weight for layer l, N_l is the number of feature maps in layer l, and M_l is the size of each feature map.
Total Loss: The total loss is a weighted sum of the content and style losses:

$$\mathcal{L}_{\text{total}} = \alpha \cdot \mathcal{L}_{\text{content}} + \beta \cdot \mathcal{L}_{\text{style}}$$

We set $\alpha = 1$ and $\beta = 1 \times 10^6$ to emphasize the style transfer while preserving the content structure.

Optimization: We use the Adam optimizer to minimize the total loss $\mathcal{L}_{\text{total}}$ by updating the target image I_t. The optimization process iteratively adjusts the pixels of I_t to reduce the difference between its content and the content image I_c, and its style and the style image I_s.

Implementation Details: We utilized a pre-trained VGG19 model from PyTorch's `torchvision.models` module for feature extraction. The layers used for computing the content and style losses were selected based on their effectiveness in capturing high-level representations (conv4_2 for content and conv1_1, conv2_1, conv3_1, conv4_1, conv5_1 for style). We built our implementation on top of existing PyTorch code for neural style transfer. Specifically, we used the pre-trained VGG19 model from PyTorch's torchvision library. A dataset class to load and transform pairs of rainy and non-rainy images and the implementation of the style transfer training loop, including the computation of content and style losses, and the optimization process are custom-written. The existing code provided utilities for loading pre-trained models and basic image transformation functions. Our custom implementation extended this with specific logic for handling paired image inputs and performing style transfer for de-raining tasks. For our style transfer experiments, we selected the following hyperparameters based on preliminary trials and insights from existing literature on neural style transfer: We used a learning rate of 0.003 for the Adam optimizer. This choice balances between convergence speed and stability. Higher learning rates led to divergence, while lower ones resulted in slow convergence. A mini-batch size of 1 was used, which is standard practice in style transfer tasks where each image pair (rainy and non-rainy) is processed independently. This ensures that the style transfer process is fine-tuned for each pair without averaging out the specific features. Each style transfer operation was run for 2000 steps, as this provided sufficient time for the generated image to converge towards the desired style while preventing overfitting. We set the content weight to 1 and the style weight to 1×10e6, prioritizing style transfer while maintaining content structure. We did not employ cross-validation for this task as neural style transfer typically focuses on qualitative results. Instead, we tested the model on various pairs from our dataset to ensure robustness and consistency.

3.4.2 Style Transfer using CycleGAN:

A disadvantage with the previously presented Pix2Pix GAN model is that it requires pairs of rainy and corresponding derained images. CycleGAN addresses this issue by using a large collection of images to learn representations from unpaired collections of rainy images (X) and non-rainy images (Y). This allows us to use a larger dataset for training since perfect correspondences are not necessary. The minimax loss for the transformation G that maps rainy to non-rainy images is given by:

$$\mathcal{L}_{cGAN2}(G, D_Y, X, Y) = \mathbb{E}_y[logD_Y(y)] + \mathbb{E}_x[log(1 - D_Y(G(x))] \quad (6)$$

$\mathcal{L}_{cGAN2}(F, D_X, Y, X)$ is similarly defined where F is a transformation that maps non-rainy to rainy images.

However, an issue with only using the minimax loss is that if for a given rainy image, the generator generates a natural-looking non-rainy image of a different location, the loss will still be low. To enforce consistency between the rainy and non-rainy images, a cycle consistency loss is also introduced which imposes the constraint that $F(G(x))$ must resemble the rainy image x and $F(G(x))$ must resemble the non-rainy image y. This loss is given by:

$$L_{cyc}(G, F) = \mathbb{E}_x[\|F(G(x)) - x\|_1] + \mathbb{E}_y[\|G(F(y)) - y\|_1] \quad (7)$$

The overall objective is given by:

$$G*, F* = \arg\min_{G,F}\max_{D_X,D_Y} \mathcal{L}_{cGAN2}(G, D_Y, X, Y) + \mathcal{L}_{cGAN2}(F, D_X, Y, X) + \lambda L_{cyc}(G, F) \quad (8)$$

Once the model is learnt, a given rainy image x can be converted to it's derained version by computing $G(x)$.

3.4.3 Using larger synthetic dataset

Another option we explored is to use a larger synthetic dataset having pairs of rainy images and their corresponding derained images. The Rain100H dataset provides such a synthetically generated dataset. However, the images look less natural and have rain streaks instead of raindrops that are seen in our test set. We trained the previously introduced Pix2Pix model using this synthetic dataset and report the comparison between using real and synthetic training data in Section 5.2.

In conclusion, the following distinct approaches are used in this project:

- Stable Diffusion Inpainting (Baseline)
- Diffusion Model and UNet-based segmentation.
- Pix2Pix
- Cycle GAN and Style Transfer
- Pix2Pix GAN with synthetic data
- Fine-tuned Diffusion Model

4. Datasets and Features

The datasets used for both our model and the baseline model involve the input of scenes that have rain and corresponding ground truth scenes under non-rainy conditions. It is challenging to collect real-world data, given that the images must be captured with minimal structural changes to the background. For our project, we have utilized the datasets provided in WeatherDiffusion repository. We use 1000 training samples, 20 validation samples and 249 test samples. Figure 3 3 shows a sample rainy image and Figure 4 4 shows a sample ground truth image. The diverse rain patterns and intensities in these datasets provide a robust foundation for training and evaluating our models. For data pre-processing, we normalize the images and resize them to 224x224 pixels. For style transfer using a pre-trained VGG19 network (19-layer deep convolutional neural network), early layers capture low-level features like edges and textures, while deeper layers capture high-level features like object parts and entire objects. Feature maps are extracted from these specific layers of the pretrained VGG19 network. Secondly, gram matrices are computed from the feature maps to capture the style information. The primary features for all other methods are the pixel intensities of the raw images themselves. For Pix2Pix GAN with synthetic data, we use 100 synthetic training images containing rain streaks instead of raindrops (synthetic datasets used are the Rain 100H) [11].

5. Results and Discussion

Evaluation To evaluate the performance of our proposed methods and compare with the baseline method, we use both qualitative and quantitative metrics:

Qualitative Evaluation: Visual inspection of the de-rained images will be conducted to assess the visual quality. We will compare the output of our method against the ground truth images and the output images of the baseline methods to ensure the visual fidelity of the results.

Quantitative Evaluation: We will employ several standard metrics to quantitatively evaluate the performance such as:

1. Peak Signal-to-Noise Ratio (PSNR) which measures the ratio between the maximum possible power of a signal and the power of corrupting noise, reflecting the quality of the de-rained image, The PSNR between two images, I and K, of size $M \times N$ pixels, is given by:

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{\text{MAX}^2}{\text{MSE}} \right)$$

where $\text{MAX} = 255$ and MSE is the mean squared error between the two images.

2. Structural Similarity Index (SSIM) assesses the similarity between the de-rained image and the ground truth image, focusing on structural information and perceptual quality.The SSIM between two images, I and K, is given by:

$$\text{SSIM}(I, K) = \frac{(2\mu_I\mu_K + c_1)(2\sigma_{IK} + c_2)}{(\mu_I^2 + \mu_K^2 + c_1)(\sigma_I^2 + \sigma_K^2 + c_2)}$$

where μ_I is the average pixel intensity, σ_I^2 is the variance of pixel intensity, σ_{IK} is the covariance, and c_1, c_2 are small constants. Both PSNR and SSIM metrics should be high to indicate promising results. The results of our experiments are summarized in Table 1.

5.1. Baseline Method

Firstly, we observe that the baseline approach that uses Stable Diffusion-based inpainting performs poorly in comparison to our other approaches. This is because of two reasons: When parts of the image are masked out, the diffusion model does not have information about the true content of the rainy image from those areas and simply makes use of context from the surrounding pixels to fill in. Secondly, when the segmentation mask does not entirely cover the rainy regions, the diffusion model fills in the missing region with raindrops to allow greater continuity which is contradictory to our goal.

5.2. GAN-based models (Pix2Pix and CycleGAN)

The outputs of the GAN-based models are shown in Figure 2. In terms of PSNR and SSIM, we obtain the best results with the Pix2Pix model that is trained on pairs of rainy and derained images. We observe a PSNR of 70.96 which is considerably higher in comparison to diffusion models and SSIM of 0.83. We attribute this to the fact that the objective for Pix2Pix is highly aligned with the goal of achieving

Model type	Model	PSNR (dB)	SSIM
Baseline	Stable Diffusion Inpainting	10.73	0.20
Compute-efficient model	Diffusion + UNet	21.74	0.78
Data-efficient models	VGG Style Transfer	60.19	0.70
	CycleGAN Style Transfer	70.17	0.83
	Pix2Pix with Synthetic data	69.53	0.78
End-to-end models (paired data)	**Pix2Pix GAN**	**70.96**	**0.83**
	Diffusion Model (w/o fine-tuning)	19.87	0.78
	Diffusion Model (with fine-tuning)	**24.51**	**0.80**

Table 1. SSIM and PSNR of various image deraining approaches

Figure 2. The original rainy image, the output of Pix2Pix trained on paired real-world data, on synthetic data, output of CycleGAN

high PSNR. The L1 loss in Equation 2 directly optimizes for lower mean squared error between the ground truth and generated image which increases PSNR. However, we note that the images generated by the fine-tuned diffusion model are of higher quality and appear more natural. When Pix2Pix the same model is trained with synethetic data, the performance is poorer. This is because the synthetic data contains unnatural rain streaks, leading to poor model generalization on real-world data having raindrops. The performance of CycleGAN is close to that of Pix2Pix in terms of SSIM and PSNR but visual inspection reveals that the raindrops have not been effectively removed. Since we do not used paired images, we cannot specify that the model's task is deraining. The model may learn styles corresponding to the colors, textures, shapes, etc. that are common in derained images and the transformation may introduce these styles into the rainy image.

5.3. Diffusion-based methods

In Figure 3, we first note that when the conditional diffusion model without fine-tuning is run for a limited number of timesteps, the output contains patch artifacts since each patch is processed separately leading to contrast variations. The image on the right of Figure 3 shows the improvement in quality that we achieve by using the same diffusion model in conjunction with the segmentation mask from UNet. Only patches containing rain have been replaced while the others are retained from the rainy image.

In Figure 4, we demonstrate that fine-tuning the diffusion model on our dataset and running it for a larger number of timesteps leads to a very high quality image that looks very similar to the ground truth image. We notice that the only deviation from the ground truth lies in the textural features of the image, such as signage; this trend is observed throughout the test set. This image also achieves a higher SSIM of 0.78 and PSNR of 24.51 when compared against the diffusion model without fine-tuning. Moreover, the result is much better compared to the stable diffusion inpainting baseline which highlights the benefit of a diffusion model that is specialized to the task of deraining.

5.4. Style Transfer

The results of applying the VGG-based style transfer is shown in Figure 5. The visual results demonstrate that our style transfer approach removes rain streaks while preserving important image details. The PSNR value of 60.19 and SSIM value of 0.7 indicate a decent improvement over the Baseline and the Diffusion + U-Net models. However, the de-rained image are not visually very similar to the ground truth images as style transfer incorporated other distinct styles as well into the content image. This may have resulted in de-rained images carrying additional stylistic features not present in the ground truth, making them visually distinct.

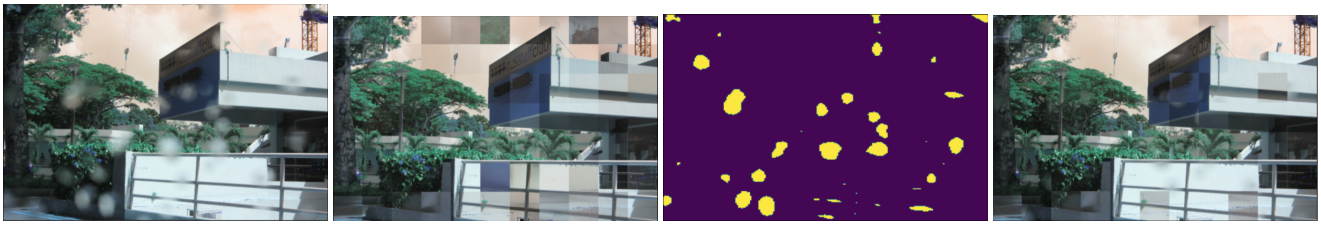

Figure 3. The original rainy image, the output of the conditional diffusion model without fine-tuning, the output of our UNet Segmentation model, and the output of U-Net+Diffusion.

Figure 4. The output from the stable diffusion baseline, from the Diffusion Model after fine-tuning, and the ground truth.

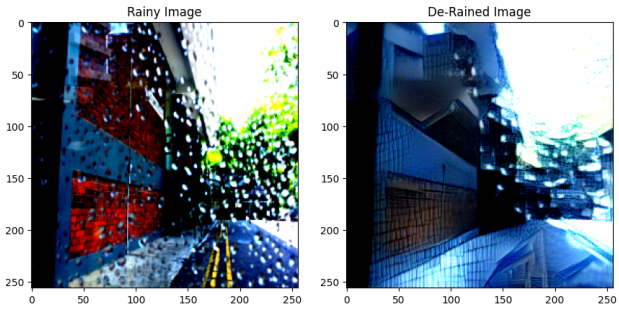

Figure 5. The original rainy image and the output from Style Transfer using VGG19

5.5. General Analysis

Based on our qualitative and quantitative comparison of the model results, we observed that the SSIM and PSNR metrics may not be the most ideal metrics for benchmarking image deraining. As observed in Section 5.2, based on the PSNR metric, it is observed that the GAN-based methods significantly outperform the diffusion-based methods; this is rather surprising given the documented superior capability of diffusion for denoising. Upon visual examination of the images, we notice that the fine-tuned diffusion model outperforms the GAN-based approaches significantly. The optimization of MSE in the loss function leads to high PSNR for GAN-based models while diffusion-based methods show better image quality. The discrepancies may also be linked to the statistical properties that vary as a result of noising and denoising.

6. Conclusion and Future Work

In our current work, we have explored various approaches to address the problem of image deraining. The key methodologies investigated include diffusion models, style transfer techniques, and generative adversarial networks (GANs). It is apparent that finetuned diffusion models yield the best qualitative result. This is followed by the GAN-based models. Style transfer produces decent results but seems to not have the utility of an expected denoising application. There are a number of trade-offs that can be made with respect to performance, such as data and compute efficiency. It must be noted that fine-tuning the diffusion model requires powerful GPUs.The use of image segmentation to speed up inference time is an example of a compute time tradeoff presented in this project, and similarly, the use of CycleGAN lends data efficiency. There are a number of avenues for improvement of results in the future given more time, resources, and compute power: Fine-tuning and improving diffusion models with advanced backbone networks and optimized hyperparameters, using hybrid models that combine CGAN and diffusion models together, or expanding/diversifying the dataset with more challenging rainy conditions.

7. Contributions

All authors have contributed equally to this report and have participated in conceptualizing and executing individual methods as described below.

- Emil: UNet+Diffusion model, Pix2Pix (on paired real-world data and synthetic rain streaks), CycleGAN style

transfer, Baseline.

- Sneha: Style transfer using pre-trained VGG, Literature review, Feature Engineering for datasets.
- Sidharth: Adaptions to the Pretrained- Diffusion Model, UNet, Finetuning the Pretrained Diffusion Model, Stable Diffusion Baseline.

References

[1] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley. Clearing the skies: A deep network architecture for single-image rain removal. *IEEE Transactions on Image Processing*, 26(6):2944–2956, 2017.

[2] K. Garg and S. K. Nayar. Vision and rain. *International Journal of Computer Vision*, 75:3–27, 2007.

[3] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[4] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[5] A. Horé and D. Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th International Conference on Pattern Recognition*, pages 2366–2369, 2010.

[6] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *Berkeley AI Research Lab*, 2018.

[7] L.-W. Kang, C.-W. Lin, and Y.-H. Fu. Automatic single-image-based rain streaks removal via image decomposition. *IEEE transactions on image processing*, 21(4):1742–1755, 2011.

[8] R. Li, L.-F. Cheong, and R. T. Tan. Heavy rain image restoration: Integrating physics model and conditional adversarial learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1633–1642, 2019.

[9] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2022.

[10] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[11] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan. Deep joint rain detection and removal from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1357–1366, 2017.

[12] W. Yang, R. T. Tan, S. Wang, Y. Fang, and J. Liu. Single image deraining: From model-based to data-driven and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[13] H. Zhang, V. Sindagi, and V. M. Patel. Image de-raining using a conditional generative adversarial network. *IEEE Computer Vision and Pattern Recognition*, 2019.