Restoring Vision in Adverse Weather Conditions with Patch-Based Denoising Diffusion Models

Ozan Ozdenizci and Robert Legenstein

The paper addresses the challenge of restoring images degraded by adverse weather conditions such as rain or snow. These ill-taken images tend to worsen the image's quality thereby losing important information. This degradation impacts many computer vision tasks such as segmentation, object detection. Deep neural networks have proven that they excel at doing this image restoration work compared to traditional methods, and this success extends to diffusion models. There are other models

1 Diffusion Model

Recently, diffusion models have provided remarkable results in various generative tasks. Therefore, this Denoising Diffusion Models based on the original diffusion model to generate output image based on noise prediction. Denoising Diffusion Probabilistic Models based on 2 Markov chain process: Forward Process and Reverse Process (t represent time step in Markov chain).

$$q(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t \mathbf{I}),$$

$$q(\mathbf{x}_{1:T} \,|\, \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}).$$

Forward Process

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

$$p_\theta(\mathbf{x}_{t-1} \,|\, \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)).$$

Reverse Process

The forward process is a process that sequentially corrupt initial image by adding Gaussian noise in each time step. The idea is that we can generate a noisy image from the original image at a particular time step of the Markov chain by following this approach:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}_t$$

Reverse process trying to conduct X0 by learning the Gaussian denoising from a completely noise image.

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

The model then is trained to try to optimize the log likelihood $\mathbb{E}_{q(\mathbf{x}_0)}[-\log p_\theta(\mathbf{x}_0)]$. The highlighted section indicates the model. It takes the noisy image at time step "t" as input and produces the predicted noise, which was added in the preceding forward step. This predicted noise is then utilized to generate the image at time step "t-1" from the image at time step "t" in a reverse process.

$$\mathbb{E}_{\mathbf{x}_0, t, \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})}\left[\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}_t, t)\|^2\right]$$

2 Denoising Diffusion Models

By combining the noisy image (which results from the forward process applied to the clean image) with the image depicting weather conditions (such as rain or snow), the model becomes capable of generating denoised images under specific weather conditions, rather than producing random outcomes:

$$p_\theta(\mathbf{x}_{0:T} \,|\, \tilde{\mathbf{x}})$$

By concatenate the noisy image (result of forward process of clean image) and the weather condition image (rain, snow), the author has embedded the condition into the model space and therefore, it can generate weather denoise image instead of random one.

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

3 Implementing

The paper employs a Unet model with the WideResNet backbone and integrates attention blocks to gather important information from different patches of the image.

Algorithm 1 Diffusive weather restoration model training

Input: Clean and weather-degraded image pairs $(\mathbf{X}_0, \tilde{\mathbf{X}})$

- **repeat**
  - Randomly sample a binary patch mask $\mathbf{P}_i$
  - $\mathbf{x}_0^{(i)} = \mathrm{Crop}(\boldsymbol{P}_i \circ \mathbf{X}_0)$ and $\tilde{\mathbf{x}}^{(i)} = \mathrm{Crop}(\mathbf{P}_i \circ \tilde{\mathbf{X}})$
  - $t \sim \mathrm{Uniform}\{1, \ldots, T\}$
  - $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  - Perform a single gradient descent step for $\nabla_\theta \| \boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0^{(i)} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, \tilde{\mathbf{x}}^{(i)}, t)\|^2$
- **until** converged
- **return** θ

Algorithm 2 Patch-based diffusive image restoration

Input: Weather-degraded image $\tilde{\mathbf{X}}$, conditional diffusion model $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \tilde{\mathbf{x}}, t)$, number of implicit sampling steps S, dictionary of D overlapping patch locations.

- $\mathbf{X}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
- **for** $i = S, \ldots, 1$ **do**
  - $t = (i-1) \cdot T/S + 1$
  - $t_{\text{next}} = (i-2) \cdot T/S + 1$ **if** $i > 1$ **else** 0
  - $\hat{\boldsymbol{\Omega}}_t = \mathbf{0}$ and $\mathbf{M} = \mathbf{0}$
  - **for** $d = 1, \ldots, D$ **do**
    - $\mathbf{x}_t^{(d)} = \mathrm{Crop}(\mathbf{P}_d \circ \mathbf{X}_t)$ and $\tilde{\mathbf{x}}^{(d)} = \mathrm{Crop}(\mathbf{P}_d \circ \tilde{\mathbf{X}})$
    - $\hat{\boldsymbol{\Omega}}_t = \hat{\boldsymbol{\Omega}}_t + \mathbf{P}_d \cdot \boldsymbol{\epsilon}_\theta(\mathbf{x}_t^{(d)}, \tilde{\mathbf{x}}^{(d)}, t)$
    - $\mathbf{M} = \mathbf{M} + \mathbf{P}_d$
  - **end for**
  - $\hat{\boldsymbol{\Omega}}_t = \hat{\boldsymbol{\Omega}}_t \oslash \mathbf{M}$ // $\oslash$: element-wise division
  - $\mathbf{X}_t \leftarrow \sqrt{\bar{\alpha}_{t_{\text{next}}}} \left(\frac{\mathbf{X}_t - \sqrt{1-\bar{\alpha}_t} \cdot \hat{\boldsymbol{\Omega}}_t}{\sqrt{\bar{\alpha}_t}} \right) + \sqrt{1-\bar{\alpha}_{t_{\text{next}}}} \cdot \hat{\boldsymbol{\Omega}}_t$
- **end for**
- **return** $\mathbf{X}_t$

4. Result

Denoising Diffusion Models have outperformed many other state-of-the-art methods, such as GANs and DDMSNet, on various datasets in terms of the two main evaluation metrics: PSNR and SSIM.

	Snow100K-S [3]		Snow100K-L [3]	
	PSNR ↑	SSIM ↑	PSNR ↑	SSIM ↑
SPANet [44]	29.92	0.8260	23.70	0.7930
JSTASR [58]	31.40	0.9012	25.32	0.8076
RESCAN [43]	31.51	0.9032	26.08	0.8108
DesnowNet [3]	32.33	0.9500	27.17	0.8983
DDMSNet [59]	34.34	0.9445	28.85	0.8772
SnowDiff$_{64}$	**36.59**	**0.9626**	**30.43**	**0.9145**
SnowDiff$_{128}$	36.09	0.9545	30.28	0.9000
All-in-One [9]	-	-	28.33	0.8820
TransWeather [7]	32.51	0.9341	29.31	0.8879
WeatherDiff$_{64}$	**35.83**	**0.9566**	**30.09**	**0.9041**
WeatherDiff$_{128}$	35.02	0.9516	29.58	0.8941

(a) Image Desnowing

	Outdoor-Rain [14]	
	PSNR ↑	SSIM ↑
CycleGAN [46]	17.62	0.6560
pix2pix [45]	19.09	0.7100
HRGAN [14]	21.56	0.8550
PCNet [53]	26.19	0.9015
MPRNet [54]	28.03	0.9192
RainHazeDiff$_{64}$	**28.38**	**0.9320**
RainHazeDiff$_{128}$	26.84	0.9152
All-in-One [9]	24.71	0.8980
TransWeather [7]	28.83	0.9000
WeatherDiff$_{64}$	29.64	**0.9312**
WeatherDiff$_{128}$	**29.72**	0.9216

(b) Image Deraining & Dehazing

	RainDrop [12]	
	PSNR ↑	SSIM ↑
pix2pix [45]	28.02	0.8547
DuRN [56]	31.24	0.9259
RaindropAttn [55]	31.44	0.9263
AttentiveGAN [12]	31.59	0.9170
IDT [6]	31.87	0.9313
RainDropDiff$_{64}$	32.29	**0.9422**
RainDropDiff$_{128}$	**32.43**	0.9334
All-in-One [9]	**31.12**	0.9268
TransWeather [7]	30.17	0.9157
WeatherDiff$_{64}$	30.71	**0.9312**
WeatherDiff$_{128}$	29.66	0.9225

(c) Removing Raindrops

Limitation: This model is based on a Markov process, which means it generates images step by step. This results in longer processing times compared to other methods. Therefore, even though the model produces excellent results, it's challenging to use it for real-time problems.