CVPR
SEATTLE, WA
JUNE 17-21, 2024

HANYANG UNIVERSITY

Visual Intelligence Lab.

LAN: Learning to Adapt Noise for Image Denoising

Changjin Kim, Tae Hyun Kim, Sungyong Baik
Hanyang University, Republic of Korea

Image Denoising

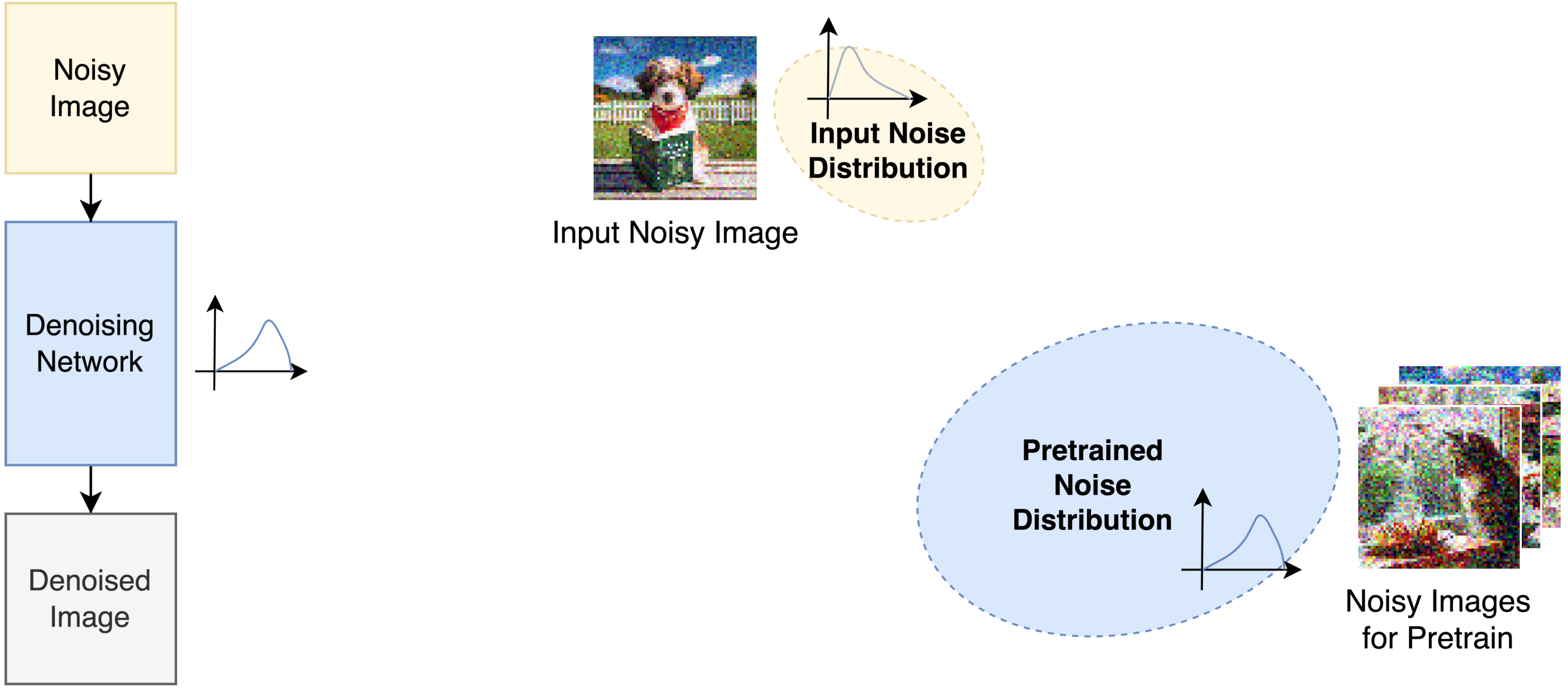
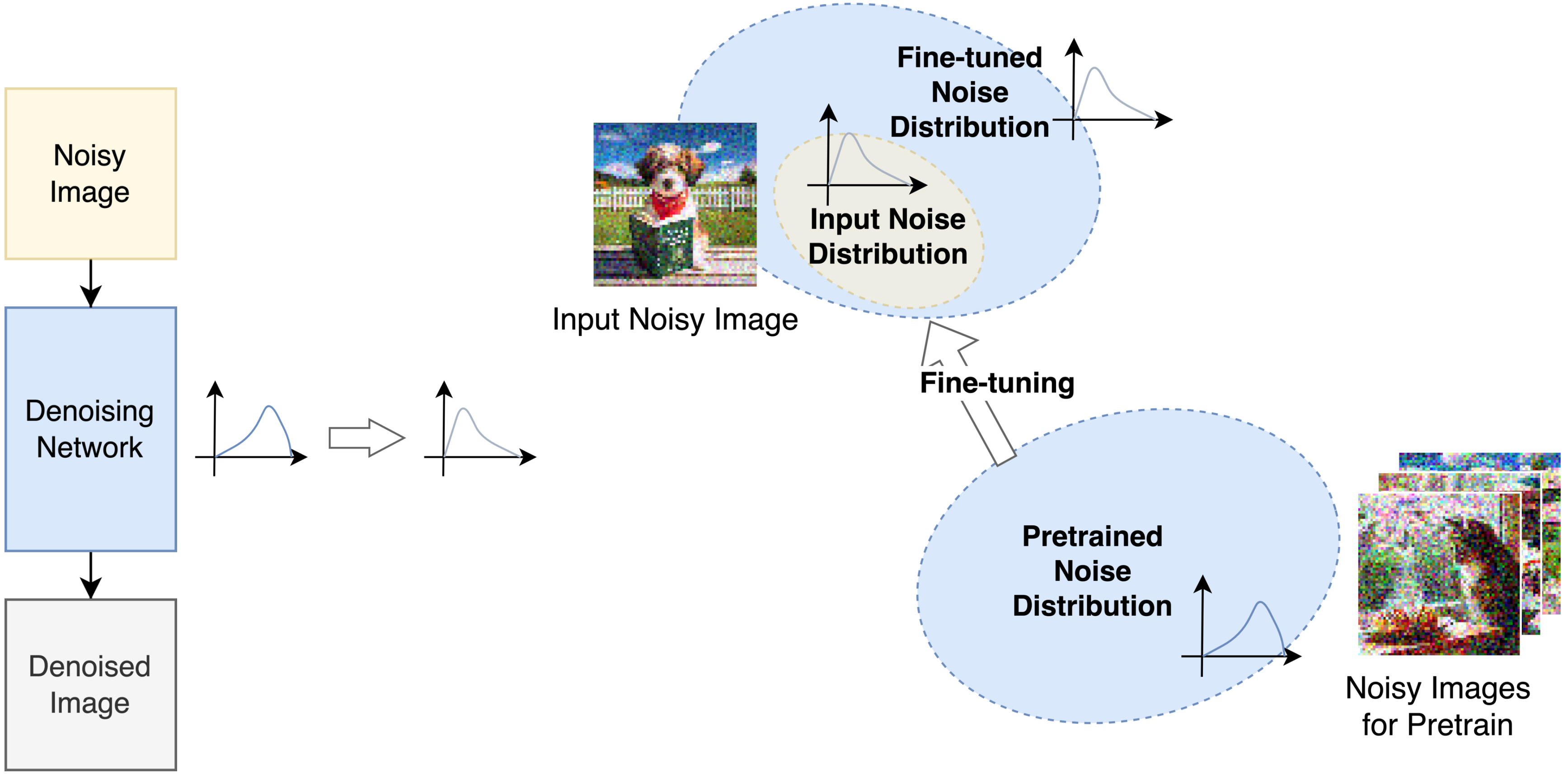
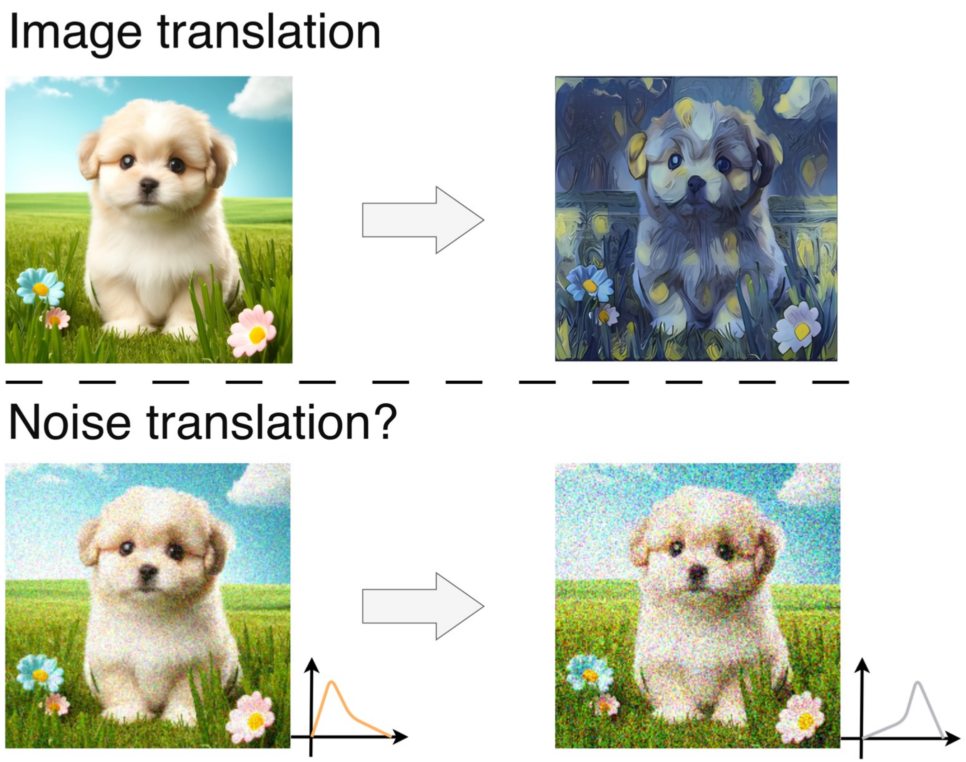


Image Denoising

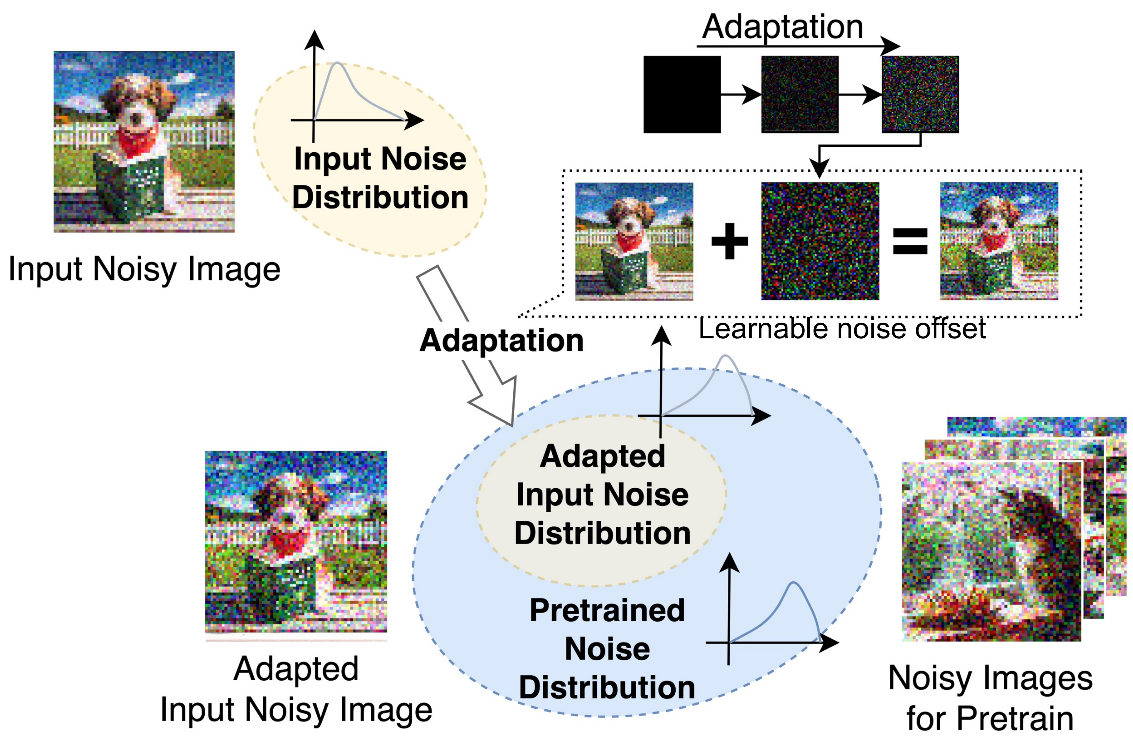


Motivation



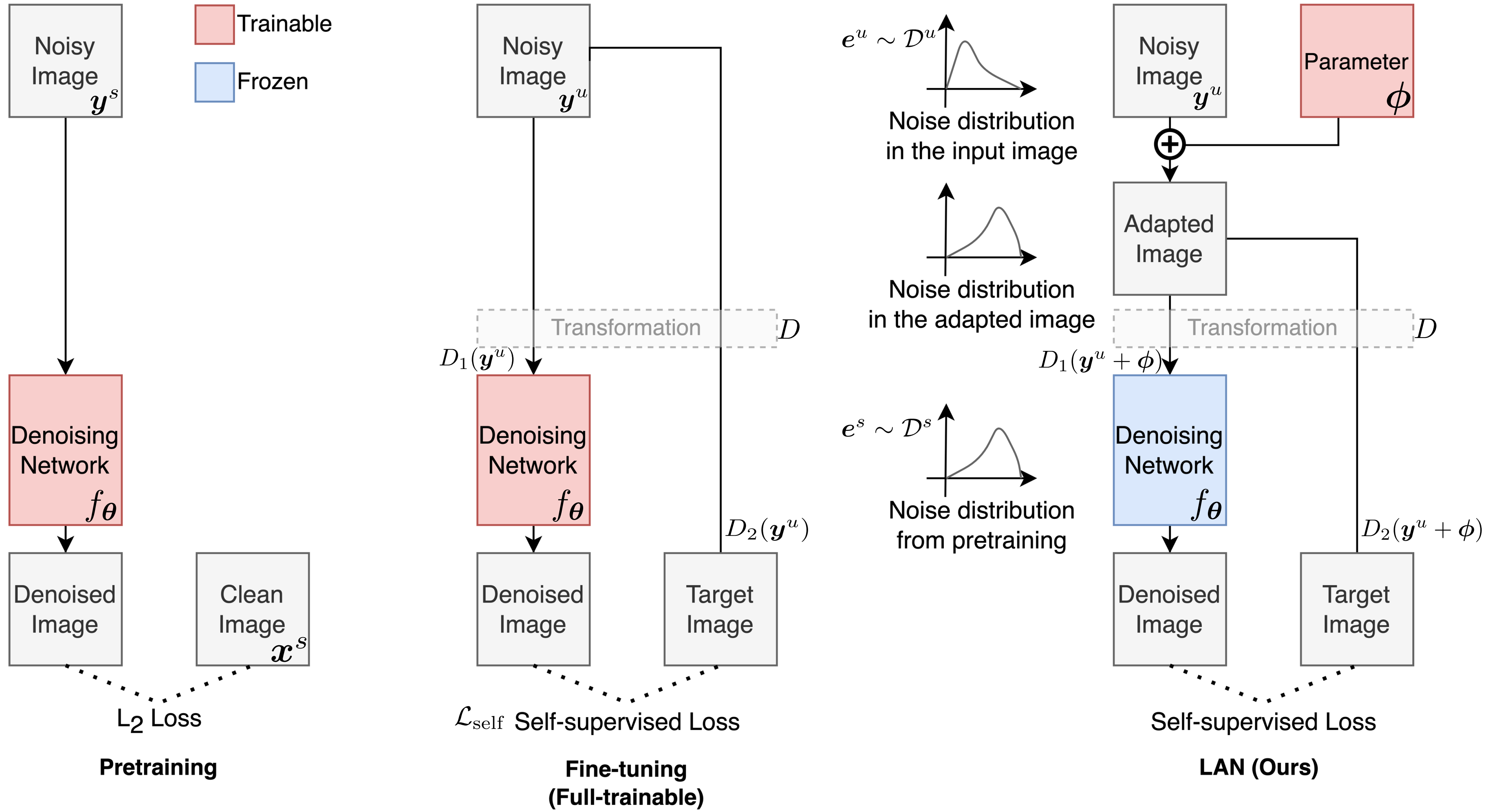
- Traditional image denoising often fails with unseen noise distributions.
- Can input noise be translated to the noise seen during training?

Overview



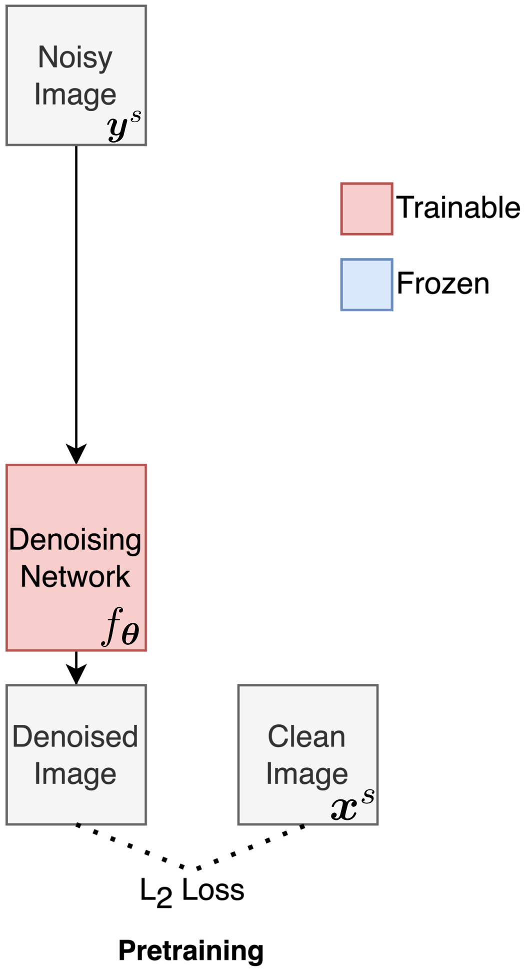
- To handle unseen noise, we propose Learning to Adapt Noise (LAN).
- LAN focuses on adapting input noise itself rather than the network.
  - Adapts input noise by adding a learnable noise offset, aligning it with the noise seen during training.

Method



Trainable
Frozen
Noisy Image $\boldsymbol{y}^s$
Denoising Network $f_{\boldsymbol{\theta}}$
Denoised Image
Clean Image $\boldsymbol{x}^s$
L2 Loss
Pretraining
Noisy Image $\boldsymbol{y}^u$
Transformation
D
$D_1(\boldsymbol{y}^u)$
$D_2(\boldsymbol{y}^u)$
Denoising Network $f_{\boldsymbol{\theta}}$
Denoised Image
Target Image
$\mathcal{L}_{\text{self}}$ Self-supervised Loss
Fine-tuning (Full-trainable)
$\boldsymbol{e}^u \sim \mathcal{D}^u$
Noise distribution in the input image
Noise distribution in the adapted image
$\boldsymbol{e}^s \sim \mathcal{D}^s$
Noise distribution from pretraining
Noisy Image $\boldsymbol{y}^u$
Parameter $\boldsymbol{\phi}$
Adapted Image
Transformation
D
$D_1(\boldsymbol{y}^u + \boldsymbol{\phi})$
$D_2(\boldsymbol{y}^u + \boldsymbol{\phi})$
Denoising Network $f_{\boldsymbol{\theta}}$
Denoised Image
Target Image
Self-supervised Loss
LAN (Ours)

Method

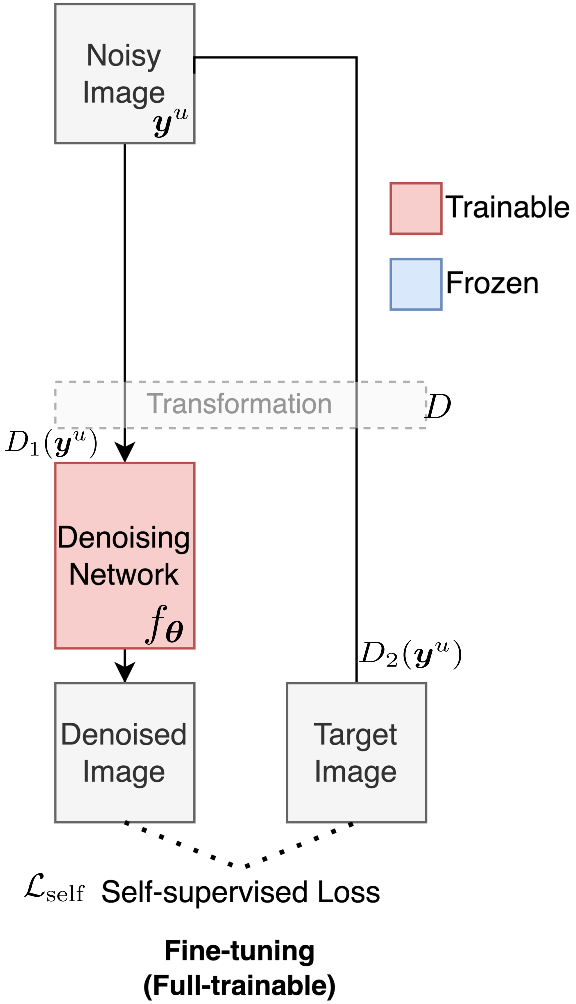


- **Pretraining:** The denoising network is pretrained on noisy-clean image pairs using L2 loss.

$$\boldsymbol{y}^s = \boldsymbol{x}^s + \boldsymbol{e}^s, \qquad \text{where} \quad \boldsymbol{e}^s \sim \mathcal{D}^s$$

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \ \mathbb{E}\left[\left\| f_{\boldsymbol{\theta}}(\boldsymbol{y}^s) - \boldsymbol{x}^s \right\|_2^2 \right]$$

Method



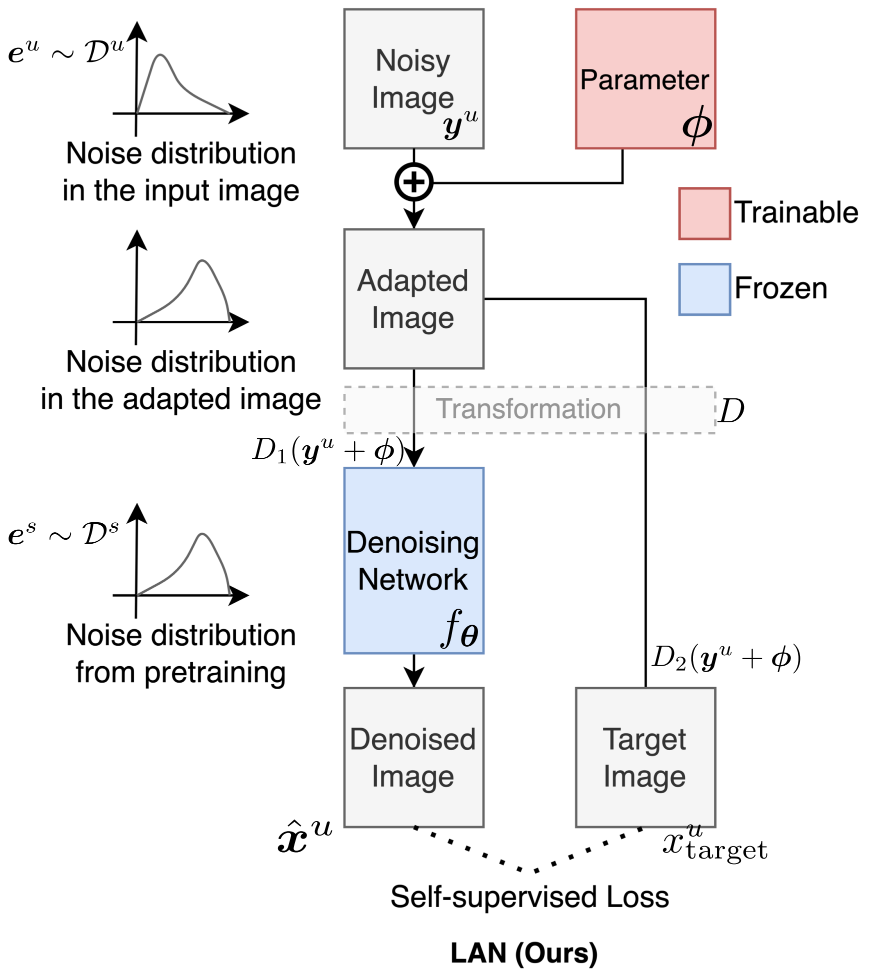
- **Fine-tuning:** The pretrained network is fine-tuned on noisy image using self-supervised loss.

$$\boldsymbol{y}^u = \boldsymbol{x}^u + \boldsymbol{e}^u, \qquad \text{where} \quad \boldsymbol{e}^u \sim \mathcal{D}^u$$

$$\Downarrow$$

$$\mathcal{L}_{\text{self}} = \left\| f_{\boldsymbol{\theta}}(D_1(\boldsymbol{y}^u)) - D_2(\boldsymbol{y}^u) \right\|_2^2$$

Method



- **LAN (Ours):** LAN adapts input noise using learnable offsets, aligning with pretrained noise distributions for better denoising.
  - Seen → Unseen

$$e^u = e^s + \epsilon^{s\to u} \qquad y^u = x^u + e^s + \epsilon^{s\to u}$$

$$y^{u\to s} := x^u + e^s = y^u - \epsilon^{s\to u}$$

  - Learning to adapt noise

$$y^{u\to s} \approx y^u + \phi$$

$$\phi^* = \arg\min_{\phi} \left\| f_{\theta^*}(D_1(y^u+\phi)) - D_2(y^u+\phi) \right\|_2^2$$

$$\hat{x}^u = f_{\theta^*}(y^u + \phi^*)$$

Experiments: Quantitative results

Model	Method	Iter.	SIDD → PolyU PSNR↑ (dB) / SSIM↑ ZS-N2N	SIDD → PolyU PSNR↑ (dB) / SSIM↑ Nbr2Nbr	SIDD → Nam PSNR↑ (dB) / SSIM↑ ZS-N2N	SIDD → Nam PSNR↑ (dB) / SSIM↑ Nbr2Nbr
DnCNN	pretrained	-	38.10 / 0.952		36.60 / 0.930	
	full-trainable	5	38.07 / 0.951	38.08 / 0.951	36.60 / 0.929	36.60 / 0.929
		10	38.04 / 0.950	38.06 / 0.951	36.59 / 0.928	36.60 / 0.928
		20	37.99 / 0.949	38.02 / 0.949	36.56 / 0.925	36.56 / 0.925
	first-layer	5	37.93 / 0.948	37.95 / 0.948	36.48 / 0.923	36.46 / 0.923
		10	37.76 / 0.943	37.83 / 0.945	36.29 / 0.915	36.28 / 0.915
		20	37.47 / 0.935	37.67 / 0.941	35.95 / 0.902	36.02 / 0.904
	last-layer	5	38.12 / 0.952	38.13 / 0.952	36.69 / 0.931	36.70 / 0.931
		10	38.13 / 0.952	38.14 / 0.952	36.75 / 0.931	36.77 / 0.931
		20	38.13 / 0.952	38.14 / 0.952	36.78 / 0.930	**36.81** / 0.930
	meta-learning	5	38.23 / **0.955**	38.23 / **0.956**	36.56 / 0.936	36.54 / **0.935**
		10	38.23 / **0.955**	38.25 / 0.955	36.66 / 0.934	36.65 / 0.934
		20	38.17 / 0.953	38.20 / 0.954	35.69 / 0.931	36.68 / 0.930
	LAN (Ours)	5	38.22 / 0.954	38.16 / 0.953	36.73 / 0.934	36.66 / 0.932
		10	**38.29** / **0.955**	38.22 / 0.954	**36.79** / 0.936	36.71 / 0.933
		20	**38.29** / **0.955**	**38.31** / **0.956**	36.78 / **0.938**	36.80 / **0.935**
Restormer	pretrained	-	39.03 / 0.966		38.03 / 0.951	
	full-trainable	5	39.09 / 0.966	39.04 / 0.965	38.14 / 0.952	38.07 / 0.951
		10	39.12 / 0.965	39.04 / 0.965	38.23 / 0.952	38.08 / 0.950
		20	39.14 / 0.965	38.98 / 0.964	38.35 / 0.953	38.05 / 0.948
	first-layer	5	39.04 / 0.965	39.00 / 0.965	38.12 / 0.951	38.07 / 0.950
		10	38.96 / 0.964	38.89 / 0.964	38.05 / 0.950	37.95 / 0.948
		20	38.74 / 0.961	38.66 / 0.961	37.65 / 0.943	37.52 / 0.941
	last-layer	5	39.07 / 0.965	39.08 / 0.965	38.09 / 0.951	38.10 / 0.951
		10	39.06 / 0.965	39.07 / 0.965	38.12 / 0.950	38.14 / 0.950
		20	39.02 / 0.964	39.03 / 0.964	38.12 / 0.948	38.14 / 0.948
	meta-learning	5	39.12 / 0.966	39.12 / 0.966	38.15 / 0.954	38.15 / 0.953
		10	39.18 / 0.966	39.13 / 0.966	38.34 / 0.954	38.21 / 0.952
		20	39.19 / 0.965	39.06 / 0.964	38.49 / 0.954	38.17 / 0.949
	LAN (Ours)	5	39.23 / 0.968	39.09 / 0.967	38.31 / 0.957	38.14 / 0.953
		10	**39.30** / **0.969**	39.14 / 0.967	38.58 / 0.961	38.25 / 0.955
		20	39.28 / **0.969**	**39.17** / **0.968**	**38.86** / **0.965**	**38.38** / **0.958**

Model	Method	Iter.	SIDD → PolyU PSNR↑ (dB) / SSIM↑ ZS-N2N	SIDD → PolyU PSNR↑ (dB) / SSIM↑ Nbr2Nbr	SIDD → Nam PSNR↑ (dB) / SSIM↑ ZS-N2N	SIDD → Nam PSNR↑ (dB) / SSIM↑ Nbr2Nbr
Uformer	pretrained	-	38.93 / 0.965		37.55 / 0.950	
	full-trainable	5	39.01 / 0.964	38.96 / 0.964	37.80 / 0.950	37.72 / 0.948
		10	39.01 / 0.963	38.92 / 0.963	37.97 / 0.950	37.77 / 0.946
		20	38.91 / 0.961	38.77 / 0.961	38.07 / 0.948	37.67 / 0.942
	first-layer	5	38.89 / 0.965	38.85 / 0.965	37.75 / 0.952	37.69 / 0.950
		10	38.82 / 0.964	38.78 / 0.964	37.76 / 0.951	37.65 / 0.948
		20	38.71 / 0.962	38.69 / 0.963	37.71 / 0.946	37.54 / 0.943
	last-layer	5	38.98 / 0.965	38.99 / 0.965	37.68 / 0.949	37.69 / 0.950
		10	39.00 / 0.965	39.01 / 0.965	37.79 / 0.949	37.81 / 0.949
		20	39.01 / 0.965	39.01 / 0.965	37.90 / 0.949	37.92 / 0.949
	meta-learning	5	39.10 / 0.967	39.09 / **0.967**	37.77 / 0.957	37.87 / 0.955
		10	39.20 / 0.966	**39.11** / 0.966	38.26 / 0.957	**38.07** / 0.952
		20	39.11 / 0.964	38.97 / 0.963	**38.52** / 0.956	38.00 / 0.947
	LAN (Ours)	5	39.12 / 0.967	39.00 / 0.966	37.82 / 0.955	37.69 / 0.951
		10	**39.21** / **0.968**	39.05 / 0.966	38.09 / 0.960	37.83 / 0.953
		20	39.20 / **0.968**	39.10 / **0.967**	38.36 / **0.964**	38.02 / **0.956**

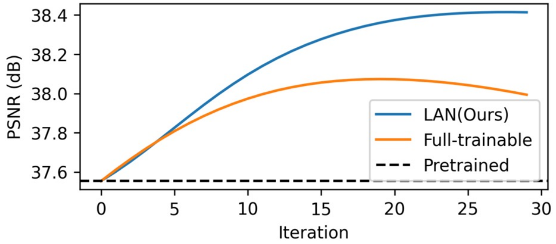
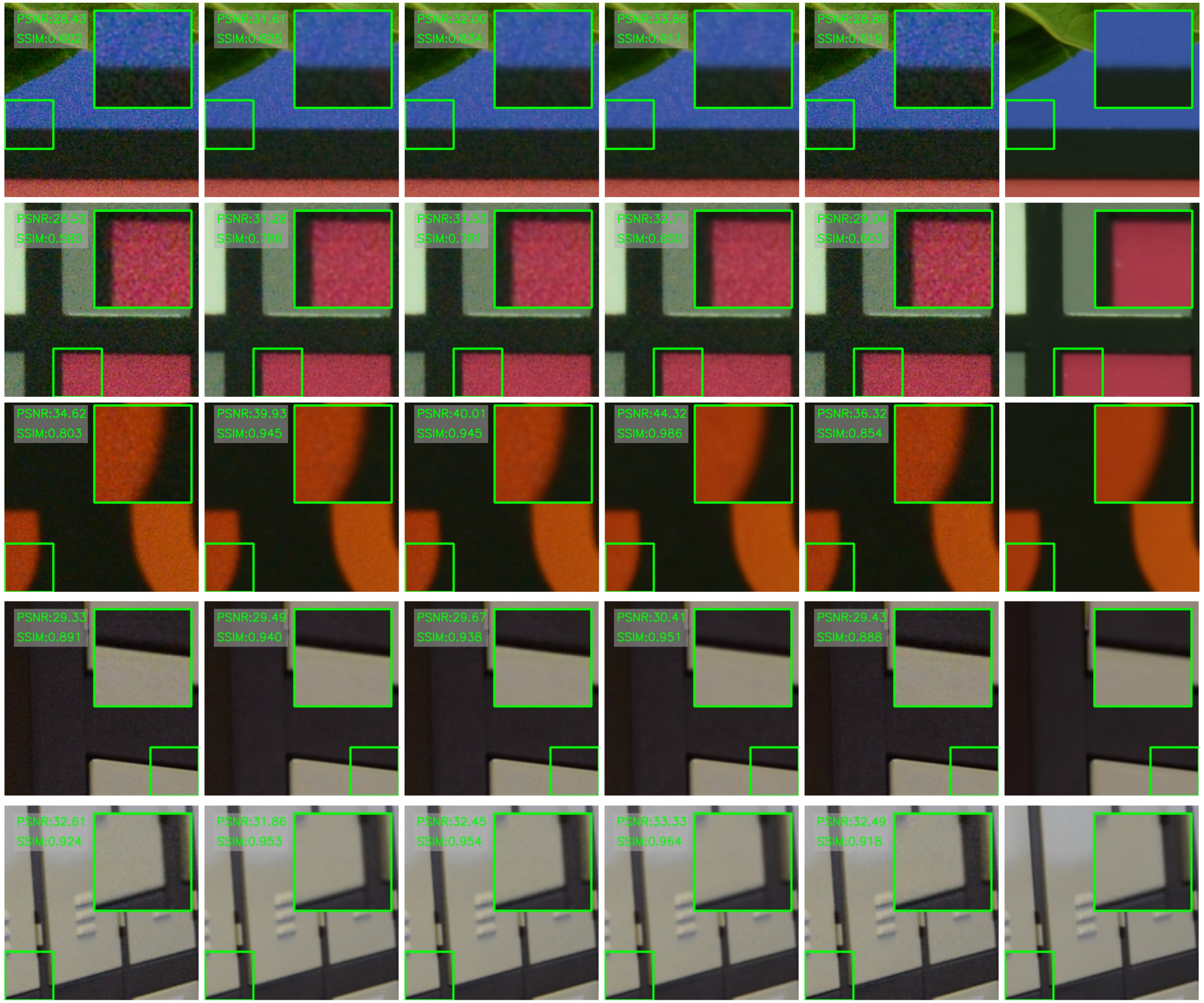
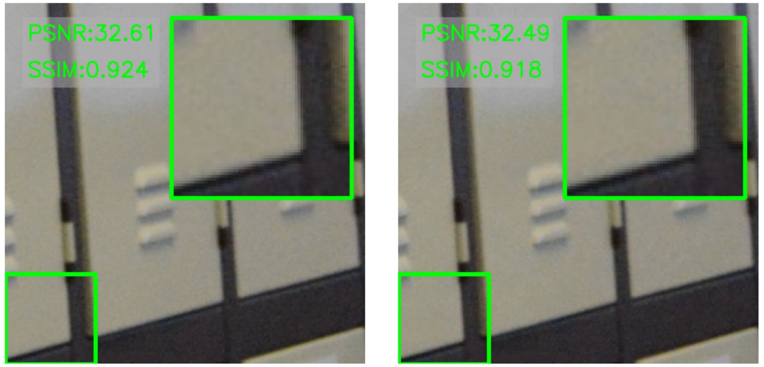


Figure 5. Plot of performance in PSNR over the number of adaptation iterations. Results are obtained with Uformer finetuned via ZS-N2N on Nam Dataset.

Experiments: Qualitative results



Noisy image | Pretrain | Full-trainable | LAN (Ours) | Adapted noisy image by LAN | Clean image



Noisy image | Adapted noisy image by LAN

Experiments: Noise adaptation

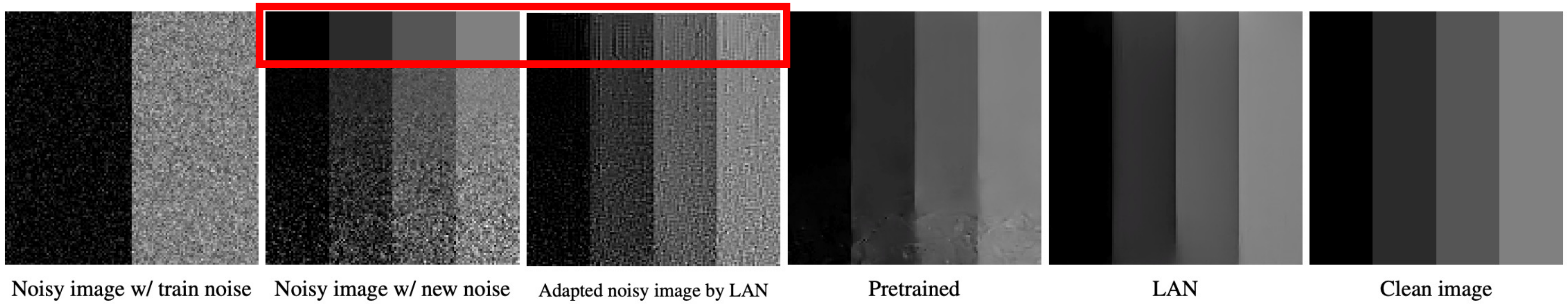


Figure 4. Visualization of synthetic noisy images. Noisy image with train noise is a noisy image that is used for pretraining a denoising network (DnCNN). Noisy image with new noise contain a new noise that is different from pretraining. Adapted noisy image by LAN is a result of noise adaptation of noisy image with new noise. We observe that noise in the adapted noisy image becomes more similar to noisy image with train noise. Particularly, we observe that noise has been added to the top of the image, where there was previously no noise. As a result, LAN helps achieve better denoising performance.

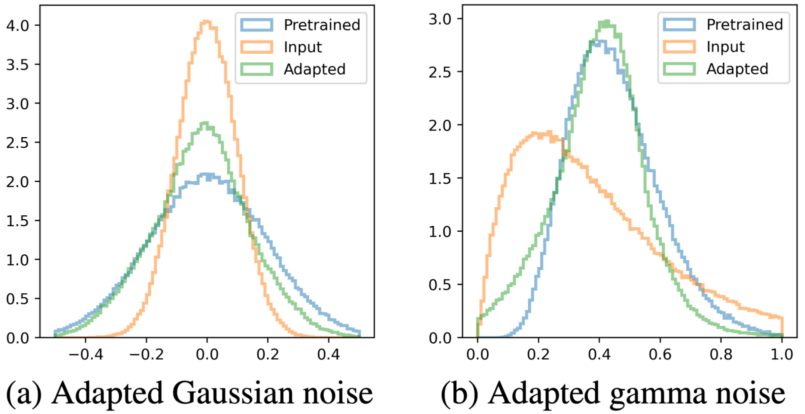


Figure 6. Histogram of synthetic noise distributions. Adapted noise distribution (green) by LAN is shown to shift the new noise distribution (orange) to already seen noise distribution (blue) during training.

Experiments: Computational Efficiency

Model	Self-loss	LAN / Full-trainable	
		Time	Memory
Restormer	ZS-N2N	79.88 %	93.27%
	Nbr2Nbr	93.04 %	92.22%
Uformer	ZS-N2N	74.10 %	73.75%
	Nbr2Nbr	85.24%	74.21%

Table 2. The runtime and memory efficiency ratio of LAN (ours) to full-trainable based on an image size of 256x256.