A WGAN-based Framework for Aerial Image Denoising under Mixed Noise Conditions

By:

Vedansh Kumar (22BCE1615)

Sarthak Chaudhary (22BCE1987)

Hrut Rajesh Shah (22BCE1760)

Problem Statement

- Aerial imagery is prone to various noise types (random, multiplicative, Gaussian).
- Existing methods usually address one noise type in isolation.
- Need for a unified denoising solution that performs across multiple noise types.

Related Works

1. The paper "Toward Convolutional Blind Denoising of Real Photographs" used a convolutional network, CBDNet, to denoise images with noise types most similar to real-world photographs. The model consisted of a noise estimator subnetwork in conjunction with a non-blind denoising subnetwork. The model was trained on both real-world images in clean-noisy pairs and synthetically noise-generated images made using realistic camera noise models from datasets such as RENOIR. The proposed model achieved PSNR and SSIM scores of 38.06 dB and 0.942 dB, respectively, on the SIDD benchmark. The key limitations were performance degradation when applied to images with different noise characteristics and heavy reliance on the accuracy of the noise detection layer.
2. The paper titled “Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising” used a feed-forward deep architecture that integrated techniques like residual learning with batch normalization, accelerating the training convergence and improving the denoising quality. The model used created by the authors (DnCNN) was designed for blind denoising, thus handling variable noise levels without prior specification. DnCNN implicitly removes the latent clean image in the hidden layers, allowing the authors to train a single DnCNN model to tackle with several general image denoising tasks, such as Gaussian denoising, single image super-resolution, and also JPEG image deblocking.

Related Works

1. Similarly, another CNN-based model, FFDNet, is a custom CNN tailored for practical image denoising tasks. FFDNet achieved this practicality as, unlike other models that required separate training for every noise intensity, FFDNet introduces a noise level map as part of the input, enabling a single model to adapt to a continuous range of noise levels. This method enhances generalisation for real-world scenarios, where noise levels vary spatially across an image. Hence, the proposed model from the paper proved to be better than other conventional CNN based models.
2. The paper aimed to tackle the problem of denoising real-world photographs with complex noise patterns using a single-stage blind denoising network known as RIDNet. The model employed a "residual on the residual" design, which allows for flow of low frequency helping in the network's ability to reconstruct the clean image alongside with a feature attention mechanism, allowing the network to focus on the most informative features among different channels. RIDNet achieved PSNR improvements of 9.5 db and 7.93 db over FFDNet and CBDNet, respectively. The effectiveness of the model however,r can diminish when applied to images with different noise characteristics as compared to the training data.

About Dataset

The dataset comprises of 4 different aerial datasets:

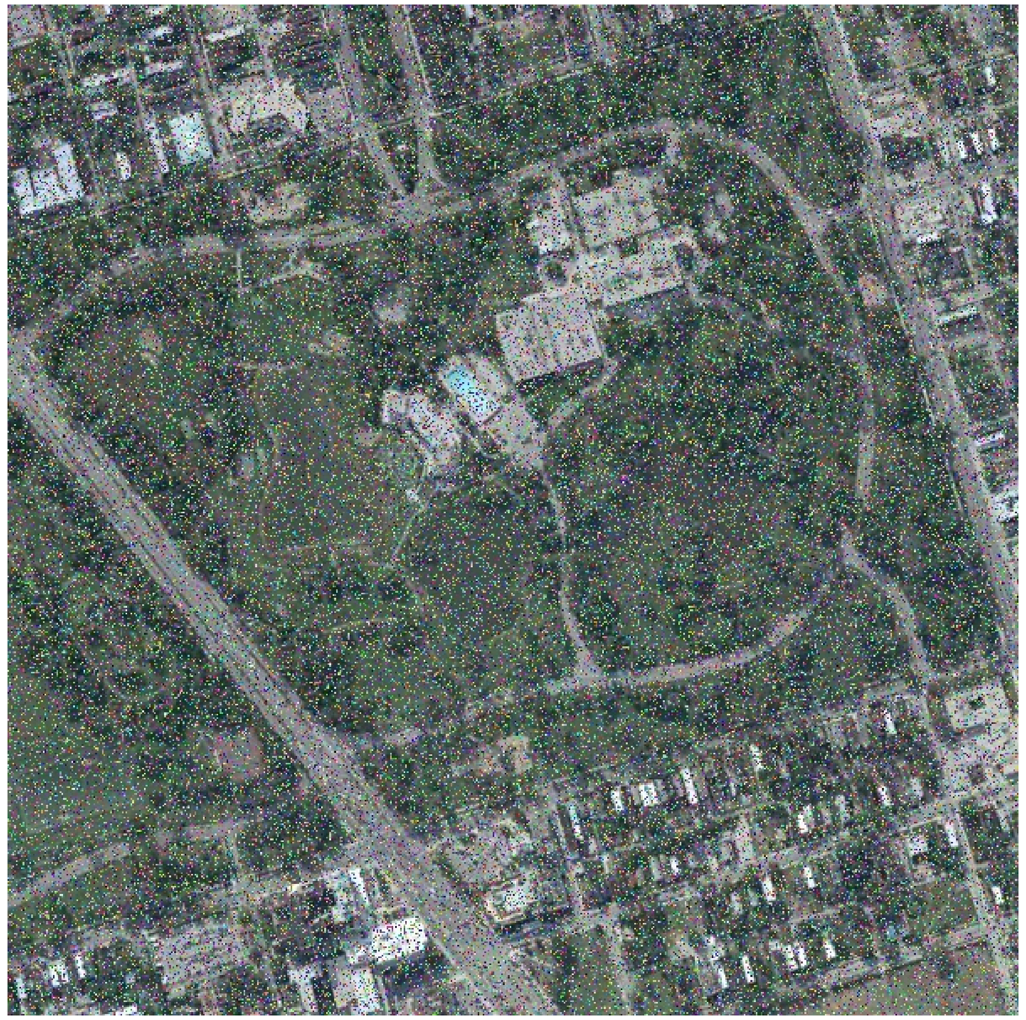
- UC-Merced dataset, contains 21 scene classes and 100 samples of size 256×256 in each class.
- WHU-RS19 dataset, has 19 different scene classes and 50 samples of size 600×600 in each class.
- RSSCN7 dataset, contains 7 scene classes and 400 samples of size 400×400 in each class.
- AID dataset, has 30 different scene classes and about 200 to 400 samples of size 600×600 in each class.

The dataset was made using introducing 3 different noises in the images which are:

- Random noise: Uniform corruption of pixels.
- Gaussian noise: Modeled as N(μ,σ2)N(μ,σ2).
- Multiplicative noise: Models speckle/texture noise common in SAR imagery.

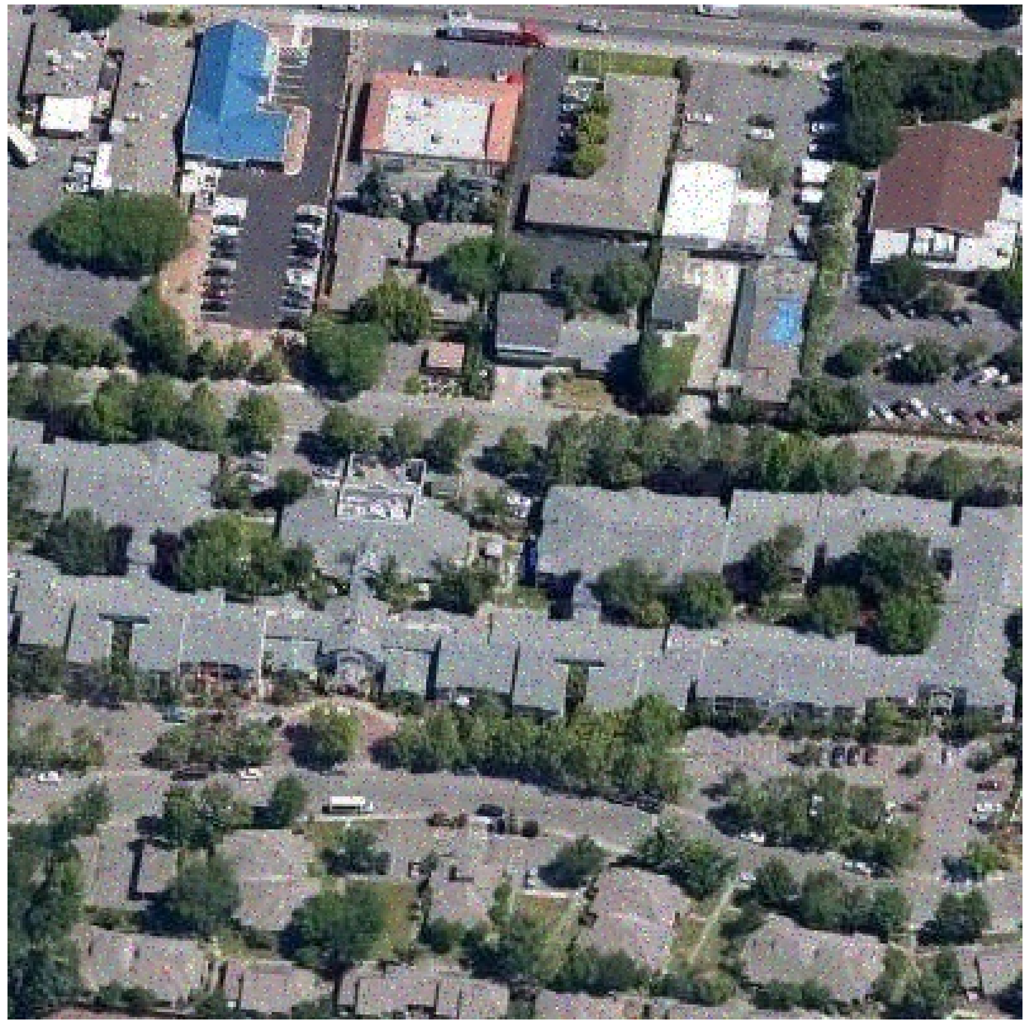
Noise Type	**Image noise**
Random	4583
Gaussian	4583
Multiplicative	4583

Example for different noise models:



Random Noise

Gaussian Noise

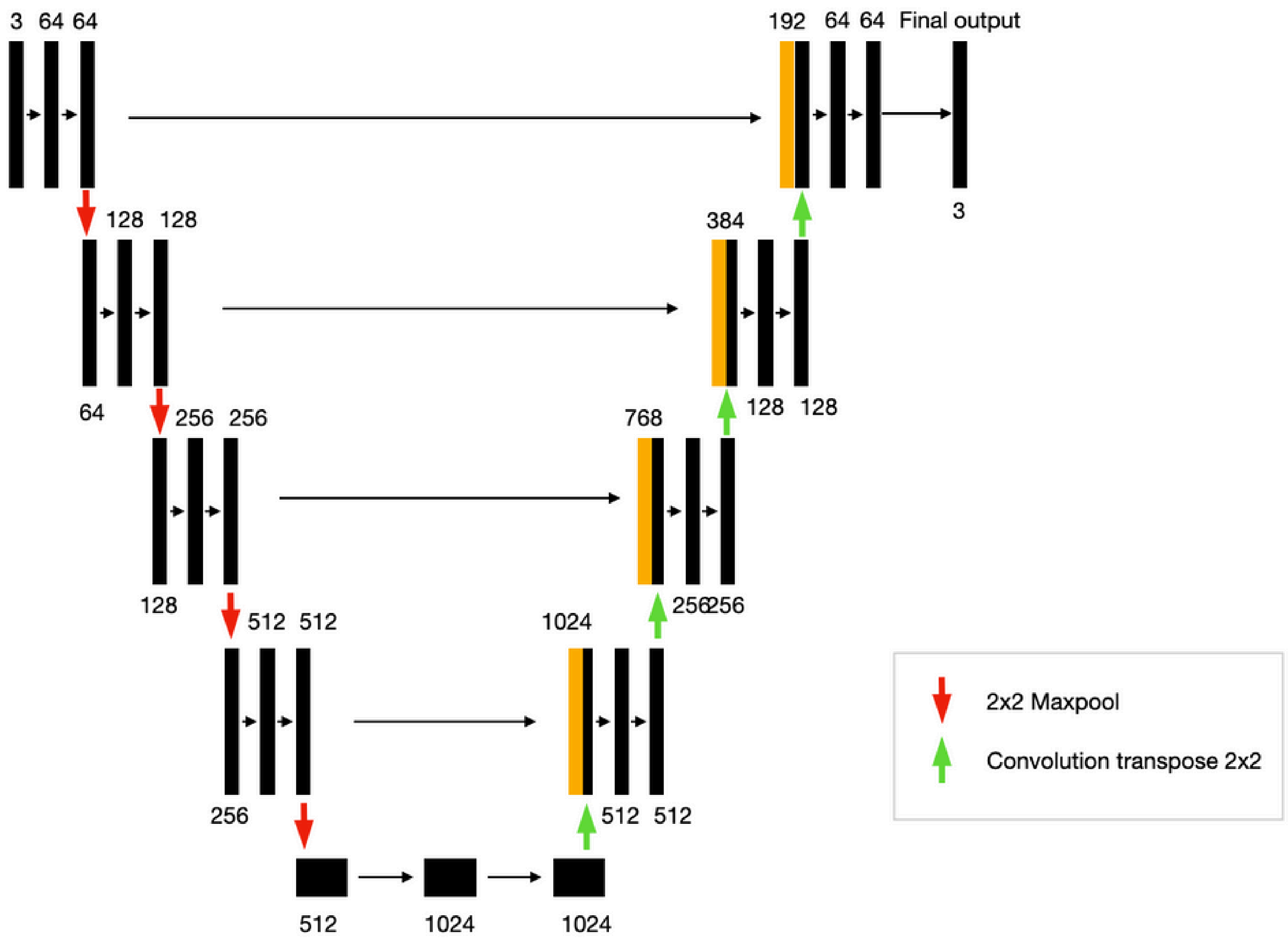


Multiplicative noise

Model architecture

Generator:

- Encoder Path
  - Stacked Conv2d + BatchNorm + ReLU layers
  - Extracts hierarchical features
  - Downsamples spatial resolution
- Decoder Path
  - ConvTranspose2d + BatchNorm + ReLU layers
  - Upsamples to original resolution
  - Skip connections from encoder preserve spatial detail
- Final output:
  - $x_{clean}=G(x_{noisy})$



Discriminator:

- Patch-based convolutional classifier
- Outputs a grid of real/fake scores
- Ensures local realism in image patches
- Architecture:
  - Conv2d → BatchNorm → LeakyReLU layers
  - Final Conv2d layer (no Sigmoid)

- The reason for **no sigmoid layer** is because we are not using vanilla GAN we are using WGAN so we are already using wassertain distance as the metric to classify image as real/fake.

Loss functions:

PERCEPTUAL LOSS

- ϕi are features extracted from a pre-trained network (e.g., VGG).
- Helps to preserve texture and high-frequency details.

$$\mathcal{L}_{\text{perceptual}} = \sum_i \|\phi_i(x) - \phi_i(\hat{x})\|_2^2$$

PIXEL-WISE RECONSTRUCTION LOSS (L1 LOSS)

- Measures absolute pixel-wise difference between real clean image x and denoised output x.
- Encourages pixel accuracy and better low-frequency structure preservation.

$$\mathcal{L}_{L1} = \mathbb{E}_{x,\hat{x}}\left[\|x - \hat{x}\|_1\right]$$

GENERATOR LOSS

- Generator tries to maximize the critic's output on fake images (i.e., make fake images look more real to the critic).
- Negative sign since it's a minimization problem.

$$\mathcal{L}_G = -\mathbb{E}_{\hat{x}\sim\mathbb{P}_g}[D(\hat{x})]$$

FINAL COMBINED LOSS

$$\mathcal{L}_G = \mathcal{L}_G^{\text{adv}} + 100\,\mathcal{L}_{\text{L1}} + 0.1\,\mathcal{L}_{\text{perc}}$$

Results:

Evaluation metrics used:

- PSNR (Peak Signal-to-Noise Ratio)
  - Measures pixel-level similarity
- SSIM (Structural Similarity Index)
  - Measures perceptual quality

Metric	**Score**
PSNR	33.04 DB
SSIM	0.9172

Clean Noisy Denoised