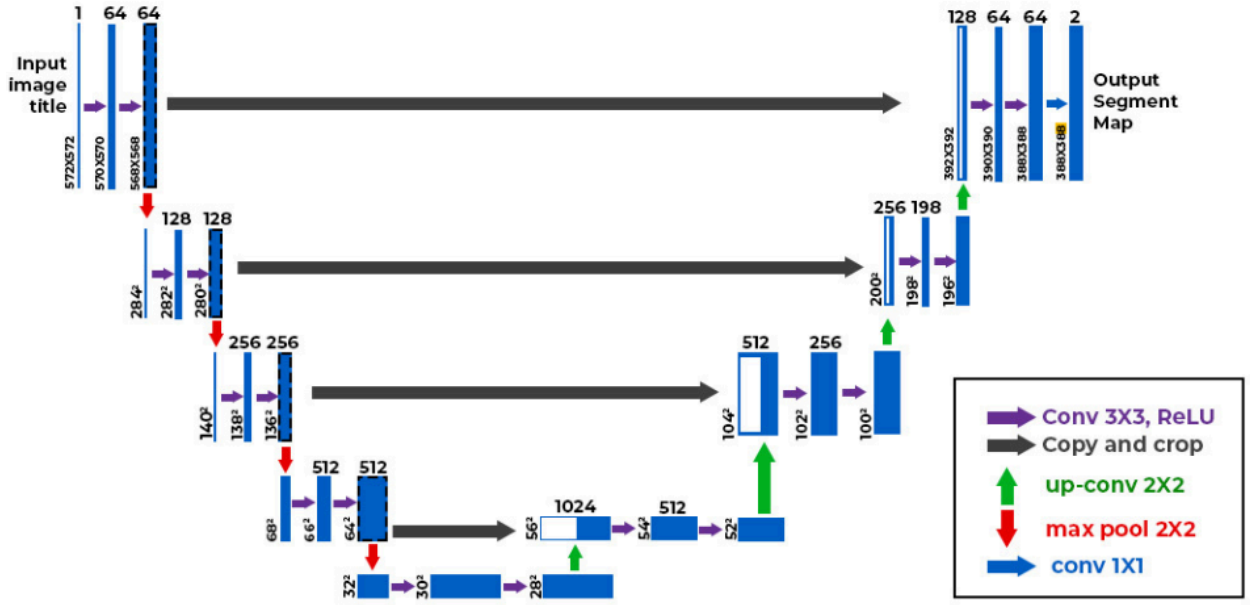
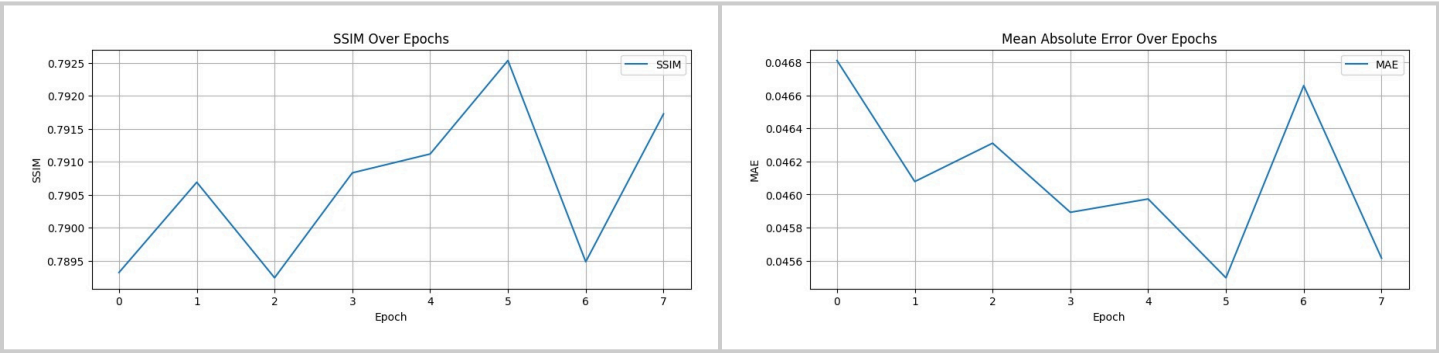
U-Net

U-Net is a deep learning model used for image segmentation. It has an encoder-decoder structure with skip connections that help preserve image details. It's widely used for tasks like cloud removal in satellite images.



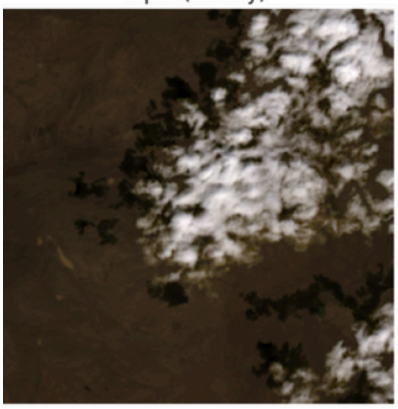
The accuracy and loss curves below provide insight into each model's convergence and generalization performance



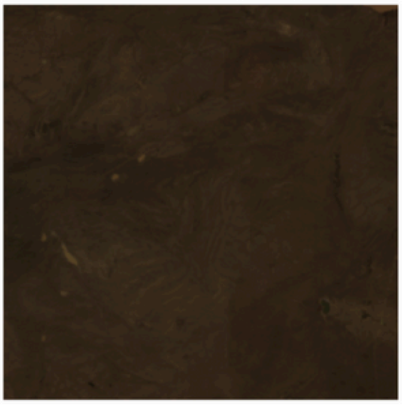
The model's output is shown below.

Sample 8

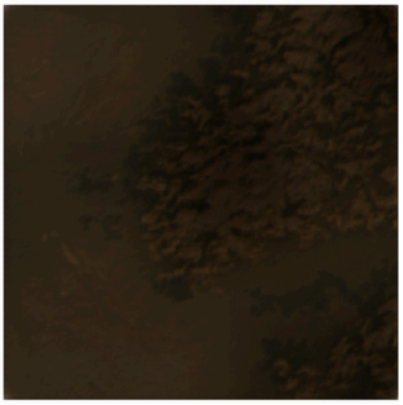
Input (Cloudy)



Ground Truth (Clear)

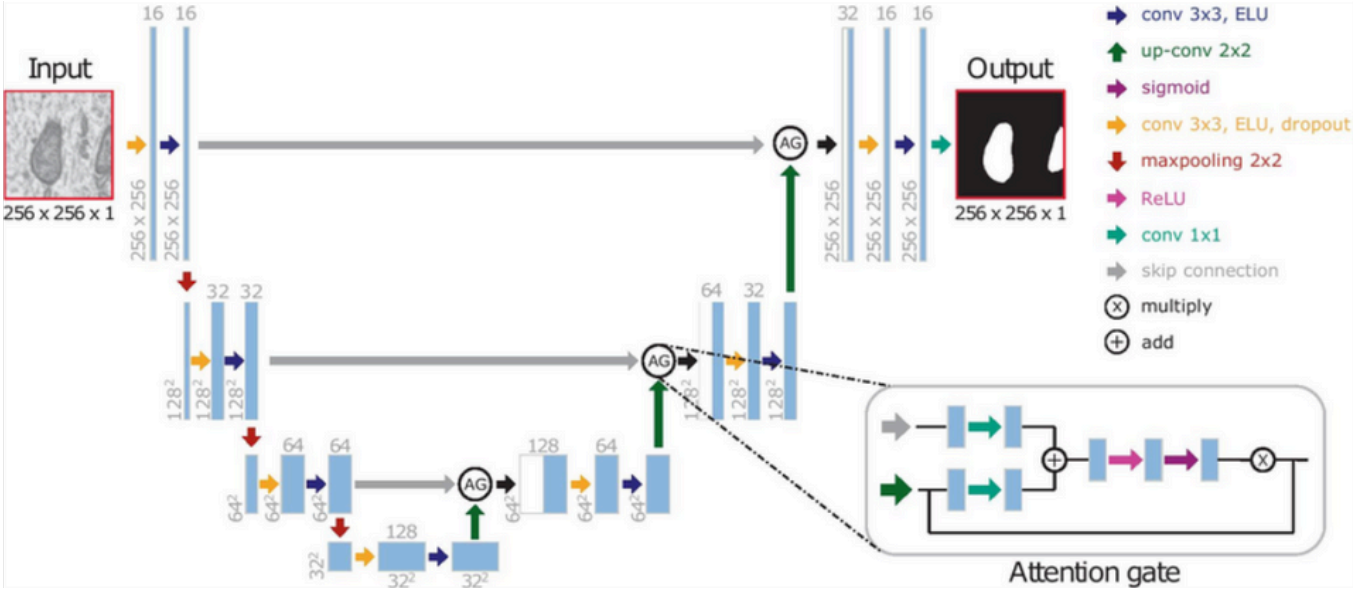


Predicted



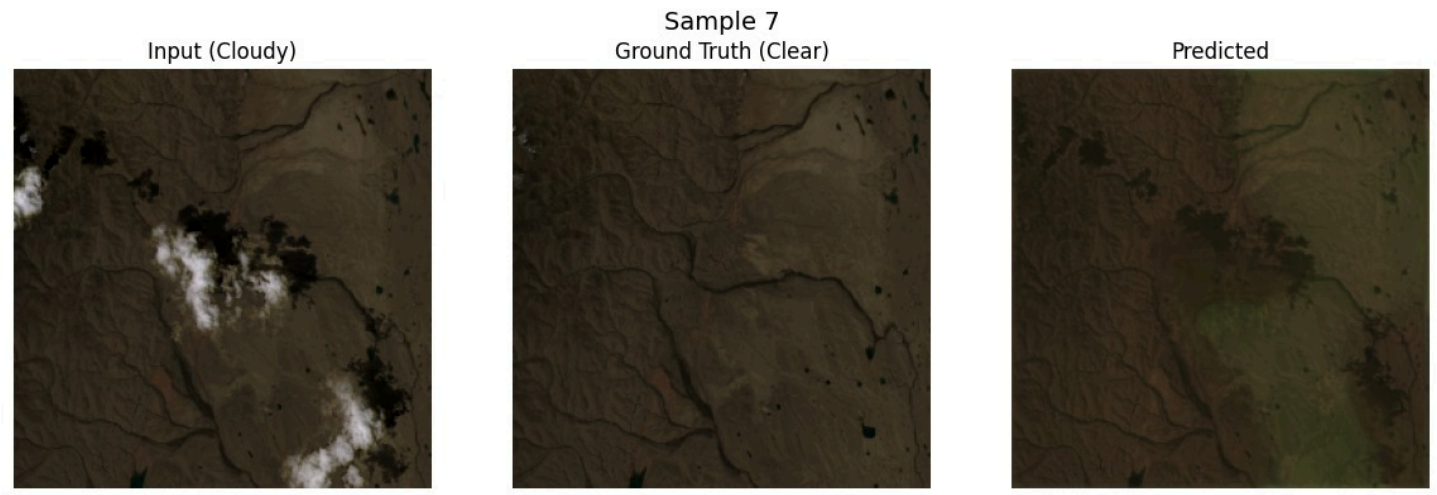
Attention U-Net

Attention U-Net is an enhanced version of the standard U-Net architecture used for image segmentation tasks. It introduces **attention gates** into the network to help the model focus on the most relevant parts of the input image, especially when dealing with complex or noisy data like satellite images or medical scans.



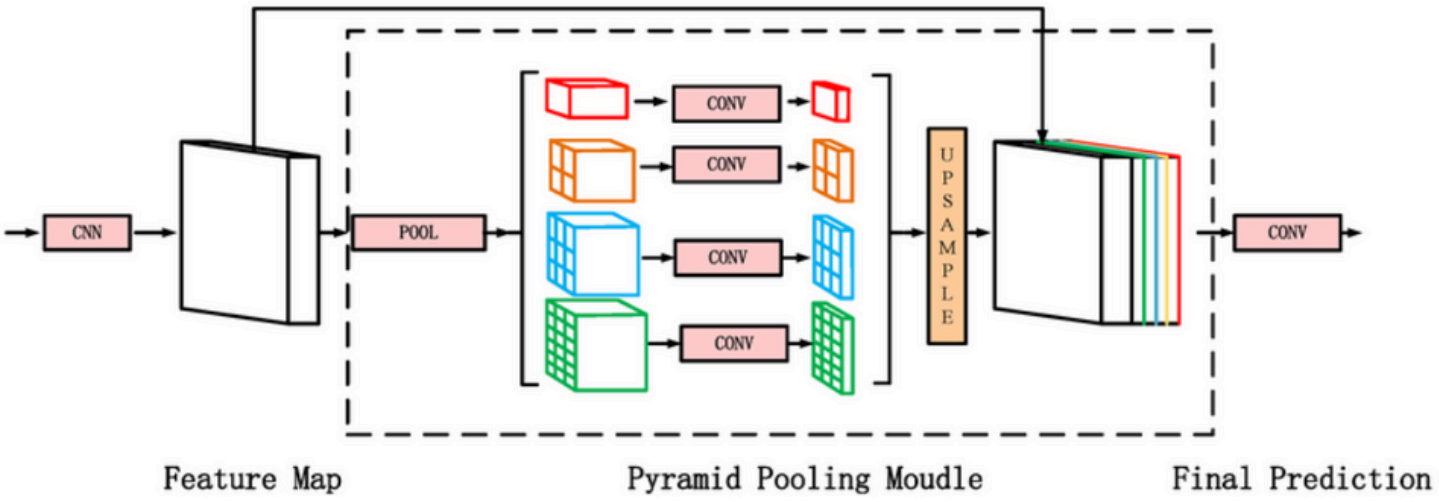
Metric	Training	Validation
Loss	0.0853	0.0775
MAE	0.0475	0.0432
SSIM	0.8263	0.8427

The model's output is shown below.

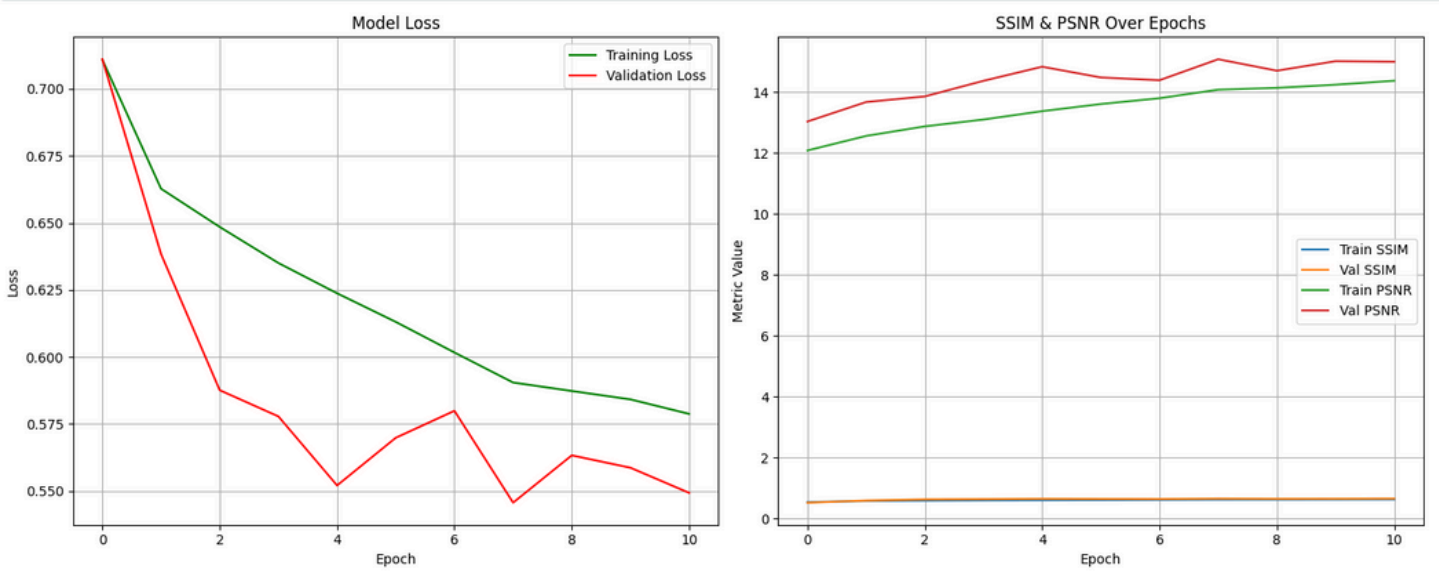


PSPNet

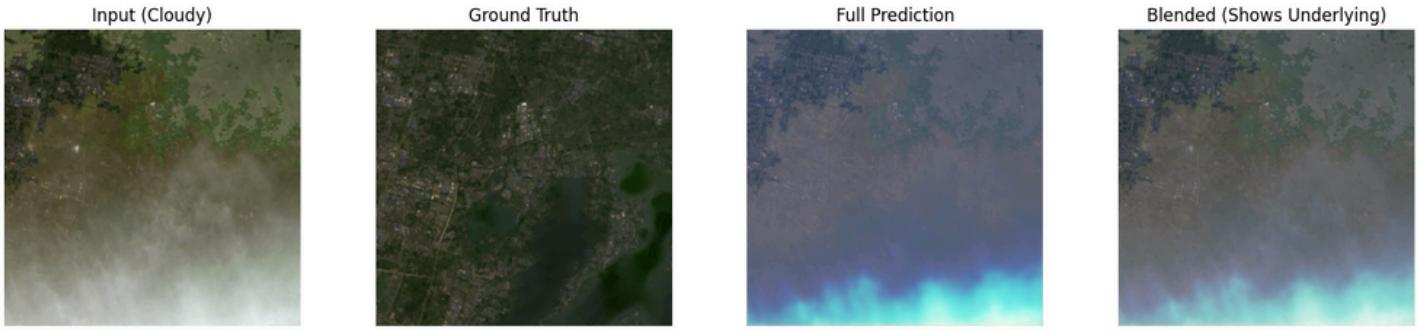
PSPNet is a segmentation model that uses **pyramid pooling** to capture features at multiple scales. This helps it understand both small and large objects, making it great for tasks like satellite image segmentation.



The accuracy and loss curves below provide insight into each model's convergence and generalization performance

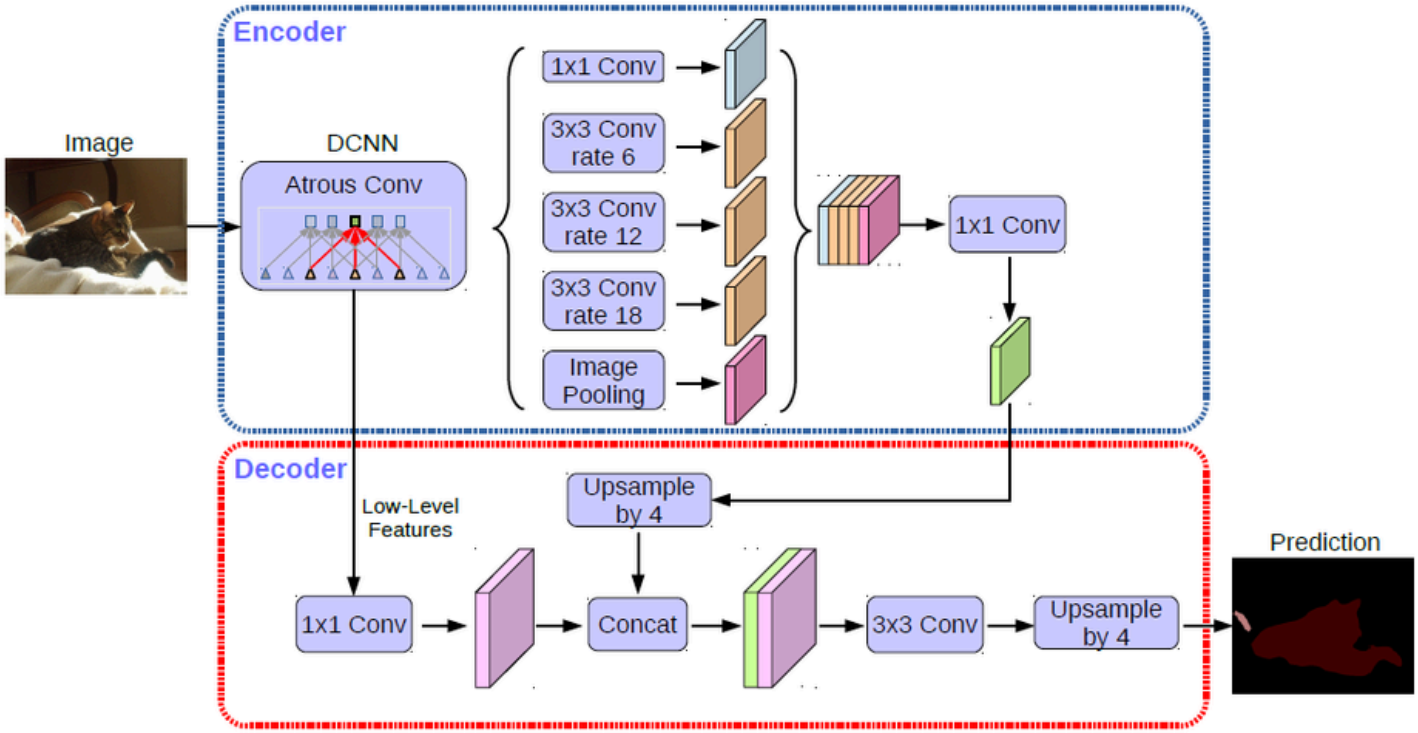


The model's output is shown below.



Attention U-Net

DeepLabv3+ is a semantic segmentation model that builds upon DeepLabv3 by adding a decoder module to enhance segmentation accuracy, particularly at object boundaries.



Metric	Training	Validation
Loss	01196	0.1189
MAE	0.1101	0.1121
SSIM	0.7324	0.7249

PIX2PIX

Expected Preprocess Input

- **Shape:** (256, 256, 4) per image.
- **Channels:**
  1. **R** = Landsat-8 Band 4 (Red)
  2. **G** = Landsat-8 Band 3 (Green)
  3. **B** = Landsat-8 Band 2 (Blue)
  4. **VV** = Sentinel-1 radar VV polarization band
- **Preprocessing Steps:**
  1. Load Landsat RGB bands + Sentinel VV into one array.
  2. Replace NaN or Inf values with safe defaults (0 or very small noise).
  3. Resize to **256×256** pixels.
  4. **Normalize per channel** to the range **[-1, 1]** (min–max scaling then scale to 2× range and shift).
- **Augmentation:** Random flips, rotations, brightness/contrast changes.
- **Batch Shape:** (batch_size, 256, 256, 4)

Expected Output

- **Shape:** (256, 256, 6) per image.
- **Channels:** Six Landsat bands
  - Likely **B1–B6** (or whichever 6 bands you stored in target TIFFs).
  - Visualization uses bands **[2, 1, 0]** from the output for RGB display, meaning:
    - Output Channel 2 → R (B4)
    - Output Channel 1 → G (B3)
    - Output Channel 0 → B (B2)
- **Value Range: [-1, 1]** during training (due to tanh output activation).
  - For visualization or saving, values are rescaled to [0, 1] via (x + 1) / 2.

Model Summary

- **Model type:** Pix2Pix (U-Net generator + PatchGAN discriminator)
- **Task:** Cloud removal — map (RGB + VV) cloudy input → (B1–B6) cloud-free output.
- **Loss:** Combined Adversarial loss (BCE) + L1 loss (MAE) with weight ratio **1 : 100**.
- **Metrics (optional):** Can add PSNR, SSIM for image quality.
- **Training input:** (256×256×4) normalized to [-1,1].
- **Training output:** (256×256×6) normalized to [-1,1].

PIX2PIX(Pretrained Weights)

Expected Preprocess Input

- **Shape:** (1024, 1024, 3) per image.
- **Channels:**
  1. **R** = Landsat-8 Band 4 (Red)
  2. **G** = Landsat-8 Band 3 (Green)
  3. **B** = Landsat-8 Band 2 (Blue)
- **Preprocessing Steps:**
  1. Load Landsat RGB bands
  2. Replace NaN or Inf values with safe defaults (0 or very small noise).
  3. Resize to 1024**×**1024 pixels.
  4. **Normalize per channel** to the range **[-1, 1]** (min–max scaling then scale to 2× range and shift).
- **Augmentation:** Random flips, rotations, brightness/contrast changes.
- **Batch Shape:** (batch_size, 1024, 1024, 3)

Expected Output

- **Shape:**(1024,1024, 3) per image.
- **Value Range: [-1, 1]** during training (due to tanh output activation).
  - For visualization or saving, values are rescaled to [0, 1] via (x + 1) / 2.

Model Summary

- **Model type:** Pix2Pix (U-Net generator + PatchGAN discriminator)
- **Task:** Cloud removal —
- **Loss:** Combined Adversarial loss (BCE) + L1 loss (MAE) with weight ratio **1 : 100**.
- **Metrics (optional):** Can add PSNR, SSIM for image quality.
- **Training input:** (1024×1024×3) normalized to [-1,1].
- **Training output:** (1024×1024×3) normalized to [-1,1].

CycleGan

Expected Preprocess Input

- **Shape:** (256, 256, 4) per image.
- **Channels:**
  1. R = Landsat-8 Band 4 (Red)
  2. G = Landsat-8 Band 3 (Green)
  3. B = Landsat-8 Band 2 (Blue)
  4. VV = Sentinel-1 radar VV polarization band
- **Preprocessing Steps:**
  1. Load Landsat RGB bands + Sentinel-1 VV band into a single 4-channel array.
  2. Replace (NaN, Inf) values with safe defaults (0 or small uniform noise).
  3. Resize all inputs to 256×256 pixels.
  4. Normalize all channels to [−1, 1] range.
  5. (Optional) Apply channel reducer: 4 → 3 channels using a 1×1 convolution before feeding to the generator.
- **Augmentation:** Random flips, rotations, brightness/contrast changes.
- **Batch Shape:** (batch size, 256, 256, 4)

Expected Output

- **Shape:** (256, 256, 3) per image (cloud-free RGB).
- **Channels:** Correspond to Landsat-8 RGB (Bands 4, 3, 2).
- **Value Range:** [-1, 1] during training (due to tanh output activation).
  - For visualization or saving, values are rescaled to [0, 1] via (x + 1) / 2.

Model Summary

- **Model Type:** CycleGAN with U-Net generators and PatchGAN discriminators.
- **Additional Module:** Channel reducer (Conv2D 1×1 layer) to map 4-channel (RGB + VV) input → 3-channel RGB.
- **Task:**
  - Map cloudy RGB + Sentinel-1 VV input → cloud-free RGB output.
  - Enforce cycle-consistency between cloudy and clean domains.
- **Generators:**
  - generator_g: Cloudy (RGB+VV) → Cloud-free RGB
  - generator_f: Cloud-free RGB → Cloudy (RGB+VV)
- **Discriminators:**
  - discriminator_x: Classifies real/fake cloudy images
  - discriminator_y: Classifies real/fake clean images
- **Loss Functions:**
  - Adversarial loss (BCE): Forces generated images to be indistinguishable from real.
  - Cycle-consistency loss (L1): Enforces $G(F(x)) \approx x$ and $F(G(y)) \approx y$.
  - Identity loss (L1): Preserves color/style when input ≈ target domain.
- **Optimizers:** Adam (learning rate = 2e−4, $\beta 1 = 0.5$).