Project Report: SAR to EO Image Translation using CycleGAN

Project 1: Summer School on Deep Dive in Deep Learning

Team Members:

1. BHAVESH BHARDWAJ
2. SATYENDRA

1. Introduction

This project explores the application of Cycle-Consistent Generative Adversarial Networks (CycleGANs) for translating Synthetic Aperture Radar (SAR) images to Electro-Optical (EO) images. The goal is to synthesize realistic EO images from SAR inputs, which can be valuable for various remote sensing applications where EO data might be unavailable or obscured (e.g., due to cloud cover).

The notebook implements a full CycleGAN pipeline, including data preprocessing, model architecture definition, training, and evaluation. It specifically targets the SEN12MS-Subset dataset, focusing on winter scenes.

2. Methodology

2.1. Data Preprocessing

The core of the data handling is the SARToEODataset class, which is responsible for loading and preparing SAR and EO image pairs.

- **Dataset Source:** The project utilizes the SEN12MS-Subset dataset, with specific directories for SAR (ROIs2017_winter_s1) and EO (ROIs2017_winter_s2) winter scenes.
- **Image Loading:** rasterio is used to load .tif image files, which are common in remote sensing for multi-band imagery.
- **Band Configuration:** The dataset loader supports different band configurations for EO images:
  - rgb: Selects Red (B4), Green (B3), and Blue (B2) bands.
  - nir_swir_red_edge: Selects Near-Infrared (B8), Short-Wave Infrared (B11), and Red Edge (B5) bands.
  - rgb_nir: Combines RGB with Near-Infrared (B8).
  - SAR images are typically 2-channel (VV and VH polarizations).
- **Normalization:** All images are normalized to the range [-1, 1], a common practice for GAN architectures to ensure stable training.
- **Transforms:** torchvision.transforms.Resize((256, 256)) is applied to all images to

standardize their dimensions.

2.2. CycleGAN Model Architecture

The CycleGAN architecture consists of two main components: Generators and Discriminators.

- **ResidualBlock:** A fundamental building block for the Generator, incorporating skip connections to facilitate gradient flow and improve performance in deep networks.
- **Generator:**
  - Composed of an initial convolution block, followed by downsampling layers, multiple residual blocks (default 9), and upsampling layers, culminating in an output layer with a Tanh activation function to produce images in the [-1, 1] range.
  - Two generators are used: G_SAR2EO (SAR to EO) and G_EO2SAR (EO to SAR).
  - Additionally, G_SAR_ID and G_EO_ID are initialized for identity mapping, although their loss component (criterion_identity) is commented out in the provided training loop.
- **Discriminator (PatchGAN):**
  - A series of convolutional layers with LeakyReLU activations and InstanceNorm2d.
  - The final layer outputs a single channel, and torch.mean(x, dim=[2, 3]) is used for classification, indicating whether a patch is real or fake.
  - Two discriminators are used: D_SAR (for real/fake SAR images) and D_EO (for real/fake EO images).

2.3. Training Configuration

The CycleGANTrainer class orchestrates the training process.

- **Loss Functions:**
  - criterion_GAN: nn.MSELoss() for adversarial loss.
  - criterion_cycle: nn.L1Loss() for cycle consistency loss.
  - criterion_identity: nn.L1Loss() for identity mapping loss (commented out in the main loss calculation).
- **Optimizers:** optim.Adam is used for all generators and discriminators, with a learning rate (lr) of 0.0002 and beta1 of 0.5.
- **Loss Weights:** lambda_cyc (cycle consistency) is set to 10.0, and lambda_id (identity) is set to 0.5.
- **Training Loop:** The train method iterates through epochs and batches, performing the train_step for each batch.
  - **Generator Training:** Calculates GAN loss (real vs. fake output of

discriminators), cycle consistency loss (reconstructing original image after two-way translation), and optionally identity loss. The total generator loss is backpropagated.
    - **Discriminator Training:** Calculates real and fake losses for each discriminator separately, aiming to distinguish real images from generated ones. The discriminator loss is backpropagated.
- **Model Saving:** Checkpoints are saved every 10 epochs, and the final model is saved at the end of training.

3. Evaluation Metrics

The project employs standard image quality metrics to evaluate the performance of the SAR-to-EO translation.

- **Denormalization:** A denormalize_image function converts images from [-1, 1] back to [0, 1] for metric calculation and visualization.
- **Structural Similarity Index (SSIM):** skimage.metrics.structural_similarity is used to measure the perceptual similarity between the generated EO images and the real EO images. SSIM scores are calculated per channel and then averaged.
- **Peak Signal-to-Noise Ratio (PSNR):** skimage.metrics.peak_signal_noise_ratio is used to quantify the quality of reconstruction.
- **Normalized Difference Vegetation Index (NDVI):** A calculate_ndvi function is provided, though it's not explicitly used in the main evaluation loop in the provided snippet. This metric is crucial for assessing the preservation of vegetation information during translation, especially when NIR bands are involved.
- **evaluate_model function:** This function puts the generator G_SAR2EO in evaluation mode, generates fake EO images from SAR inputs, and calculates the average SSIM and PSNR scores, along with their standard deviations.

4. Visualization

The visualize_results function generates and saves sample images, showing the SAR input, the real EO image, and the generated EO image for comparison. It also includes a plot_training_losses function to visualize the convergence of the different loss components over training iterations.

5. Execution and Results

The main execution section configures the data directories and initiates the training process for specified band configurations. The provided notebook runs the training for the rgb band configuration for 40 epochs.

Configuration:

- DATA_DIR: /kaggle/input/sen12ms-subset/sen12ms-subset
- SAR_DIR: DATA_DIR/ROIs2017_winter_s1
- EO_DIR: DATA_DIR/ROIs2017_winter_s2
- num_epochs: 40
- batch_size: 4

Summary of Results (RGB Configuration after 40 epochs):

- **Average SSIM:** 0.4962 ± 0.0821
- **Average PSNR:** 17.9358 ± 3.9486

The training progress, including epoch-wise average losses for Generator, Discriminators, and Cycle Consistency, is logged during execution. Sample generated images and loss plots are also produced.

6. Conclusion

This project successfully demonstrates the implementation and training of a CycleGAN model for SAR to EO image translation. The model is capable of generating EO-like images from SAR inputs, as evidenced by the SSIM and PSNR metrics. The modular design of the dataset loader and trainer allows for easy experimentation with different band configurations and hyperparameters.

Further improvements could involve:

- Experimenting with different network architectures or hyperparameter tuning.
- Incorporating the identity loss in the training to see its effect on image quality.
- Extending evaluation to include NDVI analysis for specific band configurations.
- Testing the model on different seasons or geographical regions.
- Implementing a mechanism for quantitative evaluation of generated images beyond SSIM and PSNR, possibly involving human perception studies or downstream task performance (e.g., classification on generated images).