Image Enhancement and Denoising using U-Net Architecture

-Madhu Singh(22117078)

Introduction

Image enhancement is a fundamental task in computer vision, involving the improvement of image quality to make the visual appearance clearer and more informative. It has numerous applications in areas such as medical imaging, satellite imaging, and photography. It involves the removal of noise from images to improve their visual quality and make subsequent analysis more accurate. Noise can be introduced during image acquisition due to various factors such as low-light conditions, high ISO settings, sensor artifacts, or transmission errors. The goal of image enhancement is to remove noise, increase contrast, and enhance details to facilitate better interpretation by both humans and machines.

Objectives

The primary objectives of this project are:

1. Implement a U-Net based model for enhancing grayscale images.

2. Train the model using a dataset of low-quality and high-quality image pairs.

3. Evaluate the model's performance using metrics such as Peak Signal-to-Noise Ratio (PSNR), Mean Squared Error (MSE), and Mean Absolute Error (MAE).

U-Net Architecture

The U-Net architecture, introduced by Ronneberger, O., Fischer, P., & Brox, in their paper "U-Net: Convolutional Networks for Biomedical Image Segmentation" (2015), has become a popular choice for image-to-image translation tasks. U-Net consists of an encoder-decoder structure with skip connections that directly connect corresponding layers in the encoder and decoder. This design helps the network to capture both high-level and low-level features, making it particularly effective for tasks that require precise localization, such as image segmentation and enhancement.

Results

- Test Loss: 0.0055

- Test PSNR: 23.47 dB

- Test MSE: 0.0055

- Test MAE: 0.0523

Dataset

The dataset comprises two folders:

- Train: Containing low and high subfolders with low and high-resolution training images respectively.

- Test: Containing Low and High subfolders with low and high-resolution testing images respectively.

Methodology:

1. Image Generator Function

The image generator function is responsible for reading images from the specified directory, converting them to grayscale using PIL, normalizing pixel values to [0, 1] and yielding them in batches for training and testing. This function ensures that the images are pre-processed and ready to be fed into the model.

2. PSNR Metric

PSNR (Peak Signal-to-Noise Ratio) is used to measure the quality of the enhanced images. It compares the original and enhanced images, with higher PSNR values indicating better quality. PSNR is commonly used in image processing tasks to quantify the reconstruction quality.

```python
25  # Define PSNR function
26  def psnr(y_true, y_pred):
27      max_pixel = 1.0  # Assuming that the images are normalized to [0, 1]
28      mse = tf.keras.backend.mean(tf.keras.backend.square(y_true - y_pred))
29      return 20 * tf.keras.backend.log(max_pixel / tf.keras.backend.sqrt(mse)) / tf.keras.backend.log(10.0)
```

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$
$$= 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right)$$

where , MAX_I is the maximum possible pixel value and MSE is the mean squared error between y_true and y_predicted

3. U-Net Model

The U-Net model consists of a contracting path (encoder) and an expanding path (decoder), connected by skip connections to preserve spatial information. The contracting path captures context, while the expanding path enables precise localization.
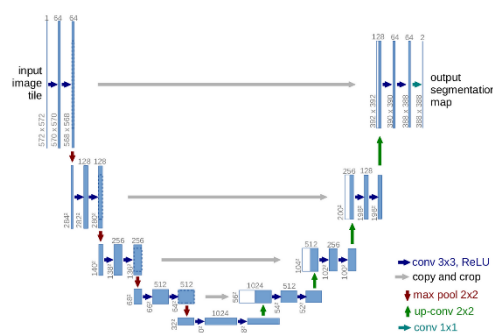
- o **Downsample (Encoder):**
  - **conv1**: Two convolutional layers with 32 filters each, followed by ReLU activation.

- **pool1**: Max-pooling layer to reduce spatial dimensions.
- **conv2**: Two convolutional layers with 64 filters each, followed by ReLU activation.
- **pool2**: Max-pooling layer to reduce spatial dimensions.

- **Upsample (Decoder)**:
  - **up1**: Transpose convolution layer to increase spatial dimensions, concatenated with corresponding encoder output.
  - **conv3**: Two convolutional layers with 32 filters each, followed by ReLU activation.
  - **up2**: Transpose convolution layer to increase spatial dimensions, concatenated with corresponding encoder output.
  - **conv4**: Two convolutional layers with 16 filters each, followed by ReLU activation.
- **Output Layer**: Convolutional layer with 1 filter and sigmoid activation to produce the enhanced image.

4. **Model Compilation and Training**

The model is compiled with the Adam optimizer for efficient training and MSE loss function to measure the reconstruction quality. It is trained using the image generators for 25 epochs with a batch size of 4.

```python
# Instantiate the U-Net model
unet = unet_model((None, None, 1))

# Compile the model
unet.compile(optimizer='adam', loss=losses.MeanSquaredError(), metrics=[psnr,
MeanSquaredError(), MeanAbsoluteError()])

# Train the model using generators
history = unet.fit(train_generator,
                   steps_per_epoch=12,
                   epochs=25,
                   validation_data=test_generator,
                   validation_steps=6)
```

5. **Model Evaluation**

The model's performance is evaluated on the test dataset using the test data generator. The evaluation metrics include test loss, PSNR , MSE and MAE.

```python
88    # Extract specific metrics from the results
89    test_loss = results[0]
90    test_psnr = results[1]
91    test_mse = results[2]
92    test_mae = results[3]
93
94    print(f"Test Loss: {test_loss:.4f}")
95    print(f"Test PSNR: {test_psnr:.2f} dB")
96    print(f"Test MSE: {test_mse:.4f}")
```

6. Training and Validation Loss Plot

The plot shows the training and validation loss over 25 epochs, indicating the model's learning progress and generalization ability.

```python
97   plt.plot(history.history['loss'])
98   plt.plot(history.history['val_loss'])
99   plt.title('Model Loss')
100  plt.xlabel('Epoch')
101  plt.ylabel('Loss')
102  plt.legend(['Train', 'Validation'], loc='upper left')
103  plt.show()
104
```
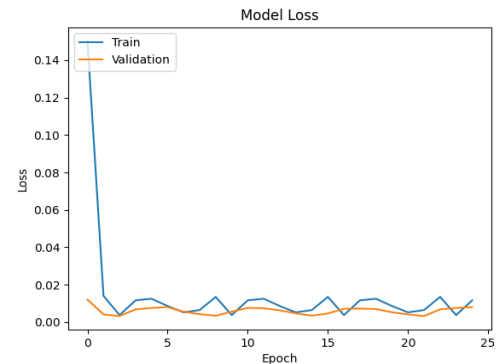

The graph demonstrates that the U-Net model used for image denoising is highly effective, as evidenced by the rapid reduction and stabilization of both training and validation losses. This indicates that the model successfully learns to denoise images and generalizes well to new data

Findings

The U-Net model effectively enhanced the quality of grayscale images, as demonstrated by the high PSNR value of 23.47 dB. The training and validation loss curves indicate that the model learned the enhancement task well with minimal overfitting.

Improvements and Future Work

1. Data Augmentation: Implementing data augmentation techniques such as rotation, flipping, and scaling can increase the diversity of the training data and improve model's validity.

2. Color Images: Extending the model to work with color images (RGB) rather than just grayscale images could broaden its applicability.

By incorporating these improvements, the model's performance can be further enhanced, making it more versatile and effective for real-world applications.

Conclusion

This project successfully demonstrates the application of the U-Net architecture for image enhancement and denoising. The model achieved promising results in terms of PSNR, MSE, and MAE, indicating its potential for improving image quality. Future work can focus on optimizing the model further and exploring advanced techniques to achieve even better performance.

References

- Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention* (MICCAI). link.

- [2404.14248] NTIRE 2024 Challenge on Low Light Image Enhancement: Methods and Results (arxiv.org)