labtask-dae-march15

March 22, 2024

[27]:
```python
import tensorflow as tf
from tensorflow.keras import layers, models
from sklearn.model_selection import train_test_split
import numpy as np
import matplotlib.pyplot as plt
import cv2
import os
```

[28]:
```python
def load_images(folder_path, target_size=(100, 100)):
    images = []
    for filename in os.listdir(folder_path):
        img = cv2.imread(os.path.join(folder_path, filename))
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB) # Convert BGR to RGB
        img = cv2.resize(img, target_size) # Resize image
        images.append(img)
    return np.array(images)
```

[29]:
```python
clean_images = load_images('/content/drive/MyDrive/train_cleaned')
noisy_images = load_images('/content/drive/MyDrive/train')
```

[30]:
```python
X_train, X_test, y_train, y_test = train_test_split(noisy_images, clean_images,␣
 ↪test_size=0.3, random_state=42)
```

[31]:
```python
def autoencoder_model():
    input_img = layers.Input(shape=(100, 100, 3))

    # Encoder
    x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(input_img)
    x = layers.MaxPooling2D((2, 2), padding='same')(x)
    x = layers.Conv2D(64, (3, 3), activation='relu', padding='same')(x)
    x = layers.MaxPooling2D((2, 2), padding='same')(x)
    encoded = layers.Conv2D(128, (3, 3), activation='relu', padding='same')(x)

    # Decoder
    x = layers.Conv2D(128, (3, 3), activation='relu', padding='same')(encoded)
    x = layers.UpSampling2D((2, 2))(x)
    x = layers.Conv2D(64, (3, 3), activation='relu', padding='same')(x)
```

```python
    x = layers.UpSampling2D((2, 2))(x)
    decoded = layers.Conv2D(3, (3, 3), activation='sigmoid', padding='same')(x)

    # Resize output to match input size
    decoded = tf.image.resize(decoded, (100, 100))



    autoencoder = models.Model(input_img, decoded)
    return autoencoder

model = autoencoder_model()
model.compile(optimizer='adam', loss='mean_squared_error')
model.summary()
```

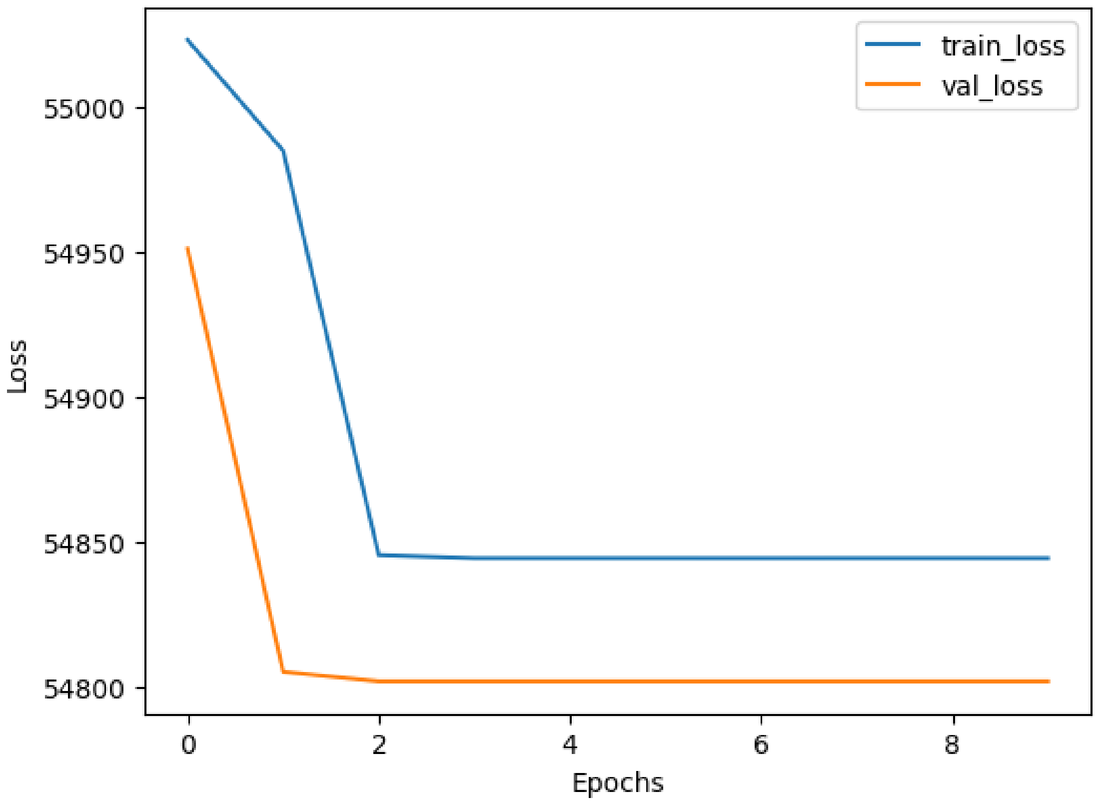
```
Model: "model_4"

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_7 (InputLayer)        [(None, 100, 100, 3)]     0

 conv2d_36 (Conv2D)          (None, 100, 100, 32)      896

 max_pooling2d_12 (MaxPooli  (None, 50, 50, 32)        0
 ng2D)

 conv2d_37 (Conv2D)          (None, 50, 50, 64)        18496

 max_pooling2d_13 (MaxPooli  (None, 25, 25, 64)        0
 ng2D)

 conv2d_38 (Conv2D)          (None, 25, 25, 128)       73856

 conv2d_39 (Conv2D)          (None, 25, 25, 128)       147584

 up_sampling2d_12 (UpSampli  (None, 50, 50, 128)       0
 ng2D)

 conv2d_40 (Conv2D)          (None, 50, 50, 64)        73792

 up_sampling2d_13 (UpSampli  (None, 100, 100, 64)      0
 ng2D)

 conv2d_41 (Conv2D)          (None, 100, 100, 3)       1731

 tf.image.resize_3 (TFOpLam  (None, 100, 100, 3)       0
 bda)
```

```
=================================================================
Total params: 316355 (1.21 MB)
Trainable params: 316355 (1.21 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

[32]:
```
history = model.fit(X_train, y_train, epochs=10, batch_size=32,␣
↪validation_data=(X_test, y_test))
```

```
Epoch 1/10
4/4 [==============================] - 12s 2s/step - loss: 55023.1758 -
val_loss: 54951.0586
Epoch 2/10
4/4 [==============================] - 11s 2s/step - loss: 54984.9531 -
val_loss: 54805.0469
Epoch 3/10
4/4 [==============================] - 12s 3s/step - loss: 54845.3555 -
val_loss: 54801.7734
Epoch 4/10
4/4 [==============================] - 18s 4s/step - loss: 54844.2812 -
val_loss: 54801.7734
Epoch 5/10
4/4 [==============================] - 12s 3s/step - loss: 54844.2812 -
val_loss: 54801.7734
Epoch 6/10
4/4 [==============================] - 10s 2s/step - loss: 54844.2812 -
val_loss: 54801.7734
Epoch 7/10
4/4 [==============================] - 11s 2s/step - loss: 54844.2734 -
val_loss: 54801.7734
Epoch 8/10
4/4 [==============================] - 12s 3s/step - loss: 54844.2812 -
val_loss: 54801.7734
Epoch 9/10
4/4 [==============================] - 12s 3s/step - loss: 54844.2812 -
val_loss: 54801.7734
Epoch 10/10
4/4 [==============================] - 10s 2s/step - loss: 54844.2734 -
val_loss: 54801.7734
```

[33]:
```
plt.plot(history.history['loss'], label='train_loss')
plt.plot(history.history['val_loss'], label='val_loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```

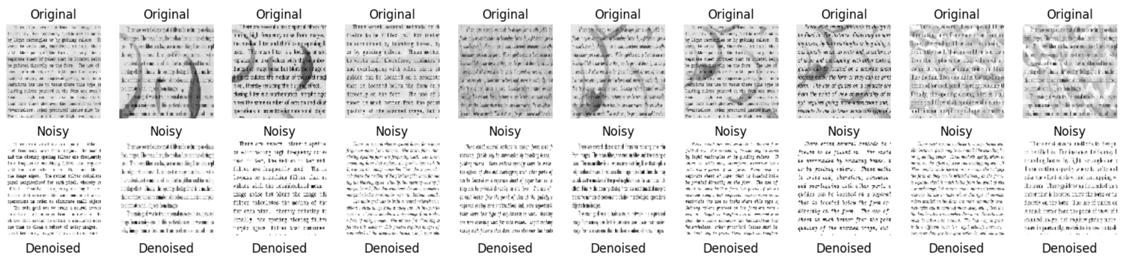


[34]:
```
decoded_imgs = model.predict(X_test)
```

```
2/2 [==============================] - 1s 248ms/step
```

[35]:
```
n = 10
plt.figure(figsize=(20, 6))
for i in range(n):
    # Original Images
    ax = plt.subplot(3, n, i + 1)
    plt.imshow(X_test[i])
    plt.title('Original')
    plt.axis('off')

    # Noisy Images
    ax = plt.subplot(3, n, i + 1 + n)
    plt.imshow(y_test[i])
    plt.title('Noisy')
    plt.axis('off')

    # Denoised Images
    ax = plt.subplot(3, n, i + 1 + 2*n)
```

```
    plt.imshow(decoded_imgs[i])
    plt.title('Denoised')
    plt.axis('off')
plt.show()
```



[36]:
```python
from sklearn.model_selection import KFold

num_folds = 5
kf = KFold(n_splits=num_folds)

fold_train_losses = []
fold_val_losses = []

for fold, (train_indices, val_indices) in enumerate(kf.split(noisy_images)):
    print(f'Fold {fold + 1}/{num_folds}:')
    X_train_fold, X_val_fold = noisy_images[train_indices], noisy_images[val_indices]
    y_train_fold, y_val_fold = clean_images[train_indices], clean_images[val_indices]

    model = autoencoder_model()
    model.compile(optimizer='adam', loss='mean_squared_error')

    history = model.fit(X_train_fold, y_train_fold, epochs=10, batch_size=32, validation_data=(X_val_fold, y_val_fold))

    fold_train_losses.append(history.history['loss'][-1])
    fold_val_losses.append(history.history['val_loss'][-1])

avg_train_loss = np.mean(fold_train_losses)
avg_val_loss = np.mean(fold_val_losses)

print(f'Average Training Loss Across Folds: {avg_train_loss}')
print(f'Average Validation Loss Across Folds: {avg_val_loss}')
```

```
Fold 1/5:
Epoch 1/10
4/4 [==============================] - 14s 3s/step - loss: 54884.5898 -
val_loss: 54632.0234
Epoch 2/10
4/4 [==============================] - 11s 3s/step - loss: 54881.5391 -
val_loss: 54632.0234
Epoch 3/10
4/4 [==============================] - 11s 3s/step - loss: 54881.5391 -
val_loss: 54632.0234
Epoch 4/10
4/4 [==============================] - 12s 3s/step - loss: 54881.5391 -
val_loss: 54632.0234
Epoch 5/10
4/4 [==============================] - 12s 3s/step - loss: 54881.5352 -
val_loss: 54632.0234
Epoch 6/10
4/4 [==============================] - 13s 3s/step - loss: 54881.5391 -
val_loss: 54632.0234
Epoch 7/10
4/4 [==============================] - 11s 3s/step - loss: 54881.5391 -
val_loss: 54632.0234
Epoch 8/10
4/4 [==============================] - 11s 3s/step - loss: 54881.5391 -
val_loss: 54632.0234
Epoch 9/10
4/4 [==============================] - 12s 3s/step - loss: 54881.5391 -
val_loss: 54632.0234
Epoch 10/10
4/4 [==============================] - 12s 3s/step - loss: 54881.5352 -
val_loss: 54632.0234
Fold 2/5:
Epoch 1/10
4/4 [==============================] - 13s 3s/step - loss: 54872.3477 -
val_loss: 54942.1484
Epoch 2/10
4/4 [==============================] - 11s 3s/step - loss: 54803.3398 -
val_loss: 54942.1484
Epoch 3/10
4/4 [==============================] - 11s 2s/step - loss: 54803.3359 -
val_loss: 54942.1484
Epoch 4/10
4/4 [==============================] - 12s 3s/step - loss: 54803.3359 -
val_loss: 54942.1484
Epoch 5/10
4/4 [==============================] - 12s 3s/step - loss: 54803.3359 -
val_loss: 54942.1484
Epoch 6/10
```

```
4/4 [==============================] - 11s 3s/step - loss: 54803.3398 -
val_loss: 54942.1484
Epoch 7/10
4/4 [==============================] - 11s 2s/step - loss: 54803.3320 -
val_loss: 54942.1484
Epoch 8/10
4/4 [==============================] - 12s 3s/step - loss: 54803.3359 -
val_loss: 54942.1484
Epoch 9/10
4/4 [==============================] - 12s 3s/step - loss: 54803.3359 -
val_loss: 54942.1484
Epoch 10/10
4/4 [==============================] - 11s 3s/step - loss: 54803.3320 -
val_loss: 54942.1484
Fold 3/5:
Epoch 1/10
4/4 [==============================] - 13s 3s/step - loss: 54941.6602 -
val_loss: 55108.2383
Epoch 2/10
4/4 [==============================] - 11s 3s/step - loss: 54772.8164 -
val_loss: 55086.4180
Epoch 3/10
4/4 [==============================] - 11s 2s/step - loss: 54766.9492 -
val_loss: 55086.4180
Epoch 4/10
4/4 [==============================] - 12s 3s/step - loss: 54766.9570 -
val_loss: 55086.4180
Epoch 5/10
4/4 [==============================] - 12s 3s/step - loss: 54766.9531 -
val_loss: 55086.4180
Epoch 6/10
4/4 [==============================] - 11s 3s/step - loss: 54766.9492 -
val_loss: 55086.4180
Epoch 7/10
4/4 [==============================] - 11s 2s/step - loss: 54766.9531 -
val_loss: 55086.4180
Epoch 8/10
4/4 [==============================] - 12s 3s/step - loss: 54766.9531 -
val_loss: 55086.4180
Epoch 9/10
4/4 [==============================] - 12s 3s/step - loss: 54766.9531 -
val_loss: 55086.4180
Epoch 10/10
4/4 [==============================] - 12s 3s/step - loss: 54766.9531 -
val_loss: 55086.4180
Fold 4/5:
Epoch 1/10
4/4 [==============================] - 13s 3s/step - loss: 54942.8438 -
```

```
val_loss: 55131.3672
Epoch 2/10
4/4 [==============================] - 11s 3s/step - loss: 54757.7773 -
val_loss: 55128.0859
Epoch 3/10
4/4 [==============================] - 11s 2s/step - loss: 54756.4453 -
val_loss: 55128.0859
Epoch 4/10
4/4 [==============================] - 12s 3s/step - loss: 54756.4492 -
val_loss: 55128.0859
Epoch 5/10
4/4 [==============================] - 12s 3s/step - loss: 54756.4453 -
val_loss: 55128.0859
Epoch 6/10
4/4 [==============================] - 11s 3s/step - loss: 54756.4453 -
val_loss: 55128.0859
Epoch 7/10
4/4 [==============================] - 11s 3s/step - loss: 54756.4453 -
val_loss: 55128.0859
Epoch 8/10
4/4 [==============================] - 12s 3s/step - loss: 54756.4453 -
val_loss: 55128.0859
Epoch 9/10
4/4 [==============================] - 12s 3s/step - loss: 54756.4453 -
val_loss: 55128.0859
Epoch 10/10
4/4 [==============================] - 12s 3s/step - loss: 54756.4492 -
val_loss: 55128.0859
Fold 5/5:
Epoch 1/10
4/4 [==============================] - 14s 3s/step - loss: 54979.8789 -
val_loss: 54351.2227
Epoch 2/10
4/4 [==============================] - 11s 3s/step - loss: 54947.1719 -
val_loss: 54351.2227
Epoch 3/10
4/4 [==============================] - 11s 2s/step - loss: 54947.1719 -
val_loss: 54351.2227
Epoch 4/10
4/4 [==============================] - 12s 3s/step - loss: 54947.1719 -
val_loss: 54351.2227
Epoch 5/10
4/4 [==============================] - 12s 3s/step - loss: 54947.1680 -
val_loss: 54351.2227
Epoch 6/10
4/4 [==============================] - 12s 3s/step - loss: 54947.1719 -
val_loss: 54351.2227
Epoch 7/10
```

```
4/4 [==============================] - 11s 3s/step - loss: 54947.1641 -
val_loss: 54351.2227
Epoch 8/10
4/4 [==============================] - 12s 3s/step - loss: 54947.1719 -
val_loss: 54351.2227
Epoch 9/10
4/4 [==============================] - 12s 3s/step - loss: 54947.1641 -
val_loss: 54351.2227
Epoch 10/10
4/4 [==============================] - 12s 3s/step - loss: 54947.1680 -
val_loss: 54351.2227
Average Training Loss Across Folds: 54831.0875
Average Validation Loss Across Folds: 54827.9796875
```

#Inference# The average training and validation losses across folds indicate the overall performance of the model. If the average losses are consistent and low across folds, it suggests that the model generalizes well to different subsets of the data, indicating robustness.