NNDL: ICP6

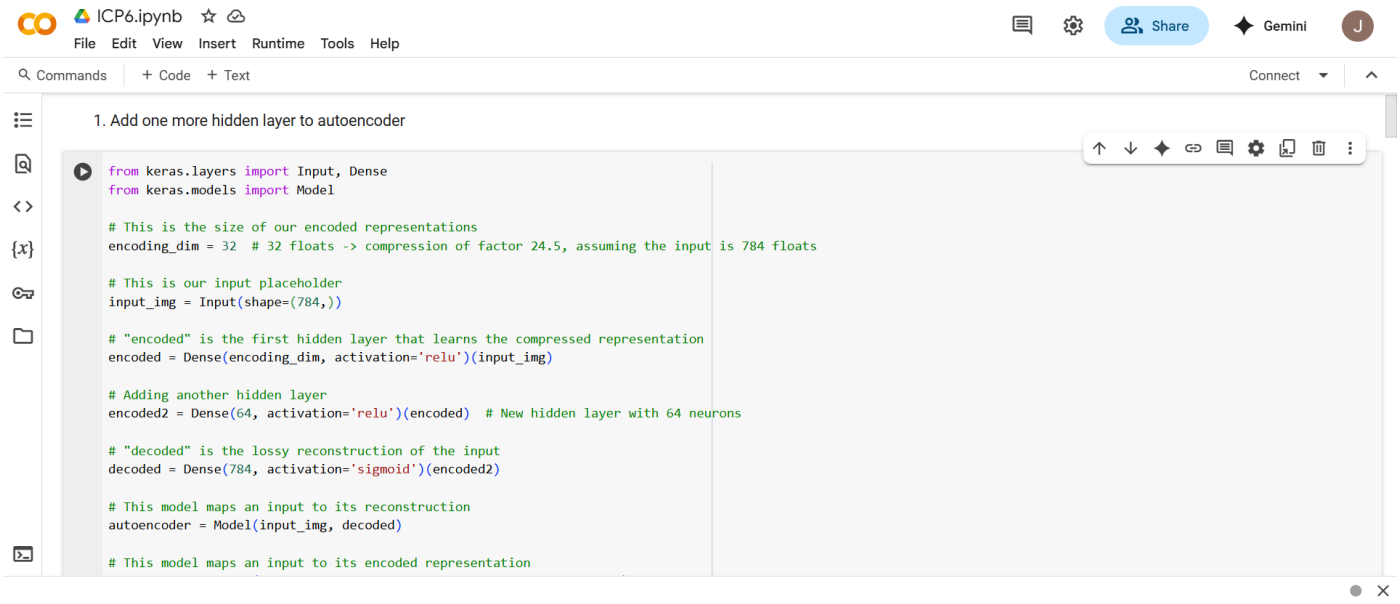
Jyothi Kiran Boddeda
700769023

GITHUB LINK : https://github.com/jyothikiranboddeda/Neural-Network-Deep-Learning.git

VIDEO LINK :

https://drive.google.com/file/d/1NCZKv3f3vv-nUEZ1Kz7WkQJmrk9x1t51/view?usp=sharing

1. Add one more hidden layer to autoencoder



2. Do the prediction on the test data and then visualize one of the reconstructed version of that test data. Also, visualize the same test data before reconstruction using Matplotlib

ICP6.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text Connect

2. Do the prediction on the test data and then visualize one of the reconstructed version of that test data. Also, visualize the same test data before reconstruction using Matplotlib

```
[ ] import matplotlib.pyplot as plt
    import numpy as np

    # Predict the reconstruction of the test data
    reconstructed_imgs = autoencoder.predict(x_test)

    # Choose an index to visualize
    index = 10  # You can change this to any index you'd like

    # Plot the original test image
    plt.figure(figsize=(10, 5))

    # Plot the original image
    plt.subplot(1, 2, 1)
    plt.imshow(x_test[index].reshape(28, 28), cmap='gray')
    plt.title('Original Image')
    plt.axis('off')

    # Plot the reconstructed image
    plt.subplot(1, 2, 2)
    plt.imshow(reconstructed_imgs[index].reshape(28, 28), cmap='gray')
```

3. Repeat the question 2 on the denoisening autoencoder

ICP6.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text Connect

3.Repeat the question 2 on the denoisening autoencoder

```
[ ] import matplotlib.pyplot as plt
    import numpy as np

    # Assuming that the model (autoencoder) has already been trained with the noisy data

    # Step 2: Make predictions on the noisy test data
    reconstructed_imgs = autoencoder.predict(X_test_noisy)

    # Step 3: Choose an index to visualize
    index = 10  # You can choose any index from the test data

    # Plot the noisy test image, original (clean) image, and reconstructed image

    plt.figure(figsize=(15, 5))

    # Plot the noisy test image
    plt.subplot(1, 3, 1)
    plt.imshow(X_test_noisy[index].reshape(28, 28), cmap='gray')
    plt.title('Noisy Image')
    plt.axis('off')

    # Plot the original (clean) test image
    plt.subplot(1, 3, 2)
    plt.imshow(X_test[index].reshape(28, 28), cmap='gray')
    plt.title('Original Image')
    plt.axis('off')

    # Plot the denoised/reconstructed image
    plt.subplot(1, 3, 3)
    plt.imshow(reconstructed_imgs[index].reshape(28, 28), cmap='gray')
```

ICP6.ipynb

File Edit View Insert Runtime Tools Help

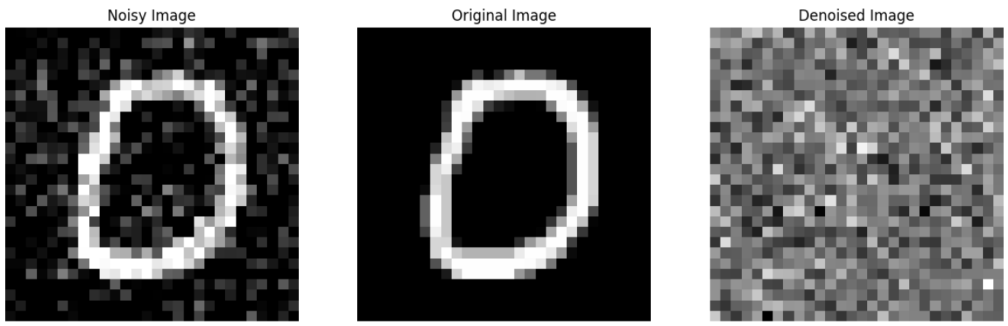
Share Gemini

Commands + Code + Text Connect

```
[ ] # Plot the denoised/reconstructed image
    plt.subplot(1, 3, 3)
    plt.imshow(reconstructed_imgs[index].reshape(28, 28), cmap='gray')
    plt.title('Denoised Image')
    plt.axis('off')

    # Show the plot
    plt.show()
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
```



4. plot loss and accuracy using the history object

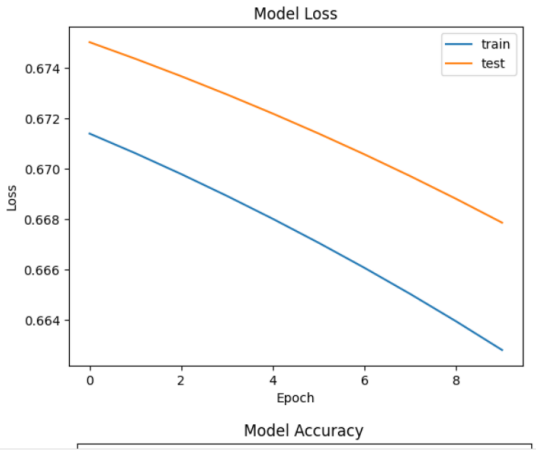
ICP6.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text Connect

```
Epoch 9/10
235/235 ━━━━━━━━━━━━━━━━━━━━ 4s 15ms/step - accuracy: 6.1246e-04 - loss: 0.6642 - val_accuracy: 0.0000e+00 - val_loss: 0.6688
Epoch 10/10
235/235 ━━━━━━━━━━━━━━━━━━━━ 4s 12ms/step - accuracy: 7.4314e-04 - loss: 0.6631 - val_accuracy: 0.0000e+00 - val_loss: 0.6679
```



ICP6.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text Connect