DS5007 Deep Learning Lab 4 - Autoencoder Implementation and Applications

Max Marks: 10
Deadline: 07/03/2025, 12:00 PM

Instructions

- Provide well-commented, indented code with meaningful variable names.
- Write the task description in separate text blocks before the corresponding code block.
- Carefully follow the task requirements and use only the specified libraries or approaches.
- Ensure all plots have appropriate axis labels, titles, and legends.
- Submit a single Jupyter Notebook (.ipynb) file named `YourName_YourRollNo_Assignment4.ipynb`.

Tasks

Task 1: Feature Extraction Using Autoencoder for Classification (5 Marks)

Task Description:

- Implement an **autoencoder** to learn a latent representation of the dataset.
- Train the autoencoder using an appropriate dataset (e.g., MNIST, Fashion-MNIST, or CIFAR-10).
- Extract the encoded representation and use it for classification.
- Compare the classification accuracy:
  - **Without pretraining:** Directly train a classifier on the original data.
  - **With pretraining:** Train a classifier using the autoencoder's learned representation.
- Fine-tune the model by training the classifier on a slightly different but related dataset.

Implementation Steps:

1. Load the dataset and preprocess it (normalize, reshape, etc.).
2. Build an autoencoder using TensorFlow/Keras:
   - Encoder: Use a few convolutional or dense layers.

    - Decoder: Reconstruct the original input from the latent representation.

3. Train the autoencoder and extract the encoded features.
4. Use these features to train a classifier (e.g., simple dense network).
5. Compare classification performance with and without pretraining.
6. Fine-tune the pretrained model on a similar dataset and analyze performance improvements.
7. Visualize results (e.g., training curves, reconstructed images, accuracy comparison).

Evaluation Criteria:

- Correct implementation of autoencoder (2 Marks)
- Proper classification pipeline and comparison (2 Marks)
- Meaningful visualizations and analysis (1 Mark)

Task 2: Denoising Autoencoder for Robust Feature Learning (5 Marks)

Task Description:

- Implement a **denoising autoencoder** and demonstrate its ability to remove noise from corrupted data.
- Train the autoencoder with noisy input images but use clean images as targets.
- Evaluate how well the denoising autoencoder reconstructs clean images.
- Compare classification accuracy using the extracted features from the denoising autoencoder vs. a standard classifier.

Implementation Steps:

1. Load and preprocess a dataset (e.g., MNIST, Fashion-MNIST, or CIFAR-10).
2. Add noise to the dataset (e.g., Gaussian noise, salt-and-pepper noise).
3. Build a denoising autoencoder:
    - Encoder: Extract meaningful features from noisy data.
    - Decoder: Reconstruct clean images from noisy input.
4. Train the model and evaluate reconstruction performance.
5. Extract features from the denoising autoencoder and use them for classification.
6. Compare the classification accuracy of:
    - Standard classifier (without autoencoder features).
    - Classifier trained using denoising autoencoder features.
7. Visualize results (e.g., noisy vs. denoised images, training curves, accuracy comparison).

Evaluation Criteria:

- Proper implementation of denoising autoencoder (2 Marks)
- Effective noise removal and reconstruction performance (2 Marks)

- Classification performance analysis and visualizations (1 Mark)

Submission Guidelines

- Ensure your code is well-structured, readable, and includes comments.
- Submit a Jupyter Notebook (`.ipynb`) file with all results and outputs included.
- Ensure all plots have appropriate labels and titles.
- Late submissions will incur penalties as per course policy.

References

- Official TensorFlow Autoencoder Guide: https://www.tensorflow.org/tutorials/generative/autoencoder
- Building Autoencoders in Keras: https://blog.keras.io/building-autoencoders-in-keras.html
- Fine-Tuning and Transfer Learning with Autoencoders: https://www.tensorflow.org/tutorials/images/transfer_learning