Task: In this assignment, students will implement a Convolutional Neural Network (CNN) from scratch using Python and NumPy to classify images from the MNIST handwritten digit dataset. The use of machine learning libraries such as scikit-learn, TensorFlow, Keras, or PyTorch is strictly prohibited. Only NumPy is allowed.

Goals: The goal is to help students understand the underlying operations of CNNs, including convolution, activation functions, pooling, and backpropagation through spatial layers.

Network Structure (minimum requirement):

- One or more convolutional layers with user-defined kernel size and number of filters
- Activation functions (e.g., ReLU)
- Pooling layer (e.g., 2×2 max pooling)
- Fully connected (dense) layer
- Output: 10 neurons with softmax for classification
- Loss: cross-entropy

Implementation Requirements:

- Implement convolution, activation, and pooling manually
- Implement forward and backward propagation manually
- Train using gradient descent
- Optional (+5pts): Visualize filters or feature maps

Output:

- Performance metrics: accuracy, precision, recall
- Graphs of loss vs. epoch and accuracy vs. epoch
- Discussion of CNN performance and insights

Hand-in:

- A PDF report with results, visualizations, and explanations
- A ZIP file containing the complete NumPy-based implementation with comments

Bonus Challenge (Optional for Extra Credit +10pts):

As an extension, students may optionally apply their CNN model to the CIFAR-10 dataset, which consists of 32×32 color images in 10 classes. Students should preprocess each image as a sequence, for example by treating each row as a vector and flattening the RGB channels per row, resulting in a sequence of 32

vectors of length 96 (32 pixels × 3 channels).

The bonus task is meant to challenge students to handle:

- Higher-dimensional input vectors
- RGB input data
- A more complex dataset
  - CIFAR-10
  - Small ImageNet (created by 侯竣奇): https://drive.google.com/drive/folders/18jqIYDLT0XQu0vBelao3_2j7vHUsoLxr?usp=sharing