ME 4990/6990 02 – Spring 2025
Homework #5 – Convolutional Neural Networks

- Please submit any Python code used as a separate file in addition to a text document (Word) which provides answers to the below problems. Writing should be in complete sentences with enough detail to demonstrate a thoughtful consideration of what is asked

Keras includes the CIFAR-10 dataset, a collection of 60,000 32x32 color images. Each contains 1 of 10 different objects and the dataset is often used as a benchmark for classification schemes. The 10 different classes are: airplane, car, bird, cat, deer, dog, frog, horse, ship, and truck. In this assignment you will build a CNN to attempt to correctly classify these images.

A short template has been included with the homework that shows how to load the data from the keras dataset.

1. Check the labels on the data to see how many of each class are present in the training and test datasets. Do you foresee any issues with this data balance? Plot a few of the images as well to get a sense for what the images look like to the human eye (You may need to reverse the normalization – dividing by 255 – done in the template).

2. Build a simple CNN with a single convolution layer and a single dense hidden layer and train it over the data. Plot the evolution of the accuracy and the loss as a function of epoch. Show a confusion matrix for the best result from the training.

3. Measure how the accuracy varies as you change the size of the single convolution layer. Does there appear to be an ideal size for training?

4. Now vary the size of the dense layer and do the same analysis as for (3)

5. Using multiple convolution and pooling layers, and a single dense layer, find a combination of hyperparameters that minimizes your validation error. What are the optimum parameters you found? Plot the confusion matrix for this best case.

6. Now replace your convolution layers with one of the existing pretrained models available in keras and optimize your dense layer to produce the best possible fit. What is the validation accuracy in this case? How does it compare to the convolution layers you trained by hand?