EECE 5644 Project
CIFAR-10 - Object Recognition in Images

Team Members: Yuxuan Cai, Jiahui Zhang, Heng Zhou

Programming Language: Python

1 Question Description

CIFAR-10 is an established computer-vision dataset used for object recognition. It is a subset of the 80 million tiny images dataset and consists of 60,000 32x32 color images containing one of 10 object classes, with 6000 images per class. It was collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton.

There are 50,000 training images and 10,000 test images in the official data. For each image in the test set, the algorithm should predict a label for the given id. The labels must match the official labels exactly {airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck}.

2 Source of Data

The CIFAR-10 data consists of 60,000 32x32 color images in 10 classes, with 6000 images per class. There are 50,000 training images and 10,000 test images in the official data. Kaggle have preserved the train/test split from the original dataset.

3 Data description

Kaggle provide train and test files:

- train.7z - a folder containing the training images in png format
- test.7z - a folder containing the test images in png format
- trainLabels.csv - the training labels

To discourage certain forms of cheating (such as hand labeling) Kaggle have added 290,000 junk images in the test set. These images are ignored in the scoring. Kaggle have also made trivial modifications to the official 10,000 test images to prevent looking them up by file hash. These modifications should not appreciably affect the scoring.

The label classes in the dataset are:

- airplane
- automobile
- bird
- cat
- deer
- dog
- frog

- horse
- ship
- truck

The classes are completely mutually exclusive. There is no overlap between automobiles and trucks. "Automobile" includes sedans, SUVs, things of that sort. "Truck" includes only big trucks. Neither includes pickup trucks.

4 Outline of approach

We are going to train a neuron networks to do classification.

Before load data, we will doing a data cleaning step first. The 32×32 images have to be reshape to match the neuron networks input. we can normalize the RGB values by dividing the x_train and x_test values by 255. Also, we could convert the images to grayscale, apply discrete histogram flattening, apply de-noising filters, or separate the phase and magnitude of the images to create more features if needed. These tasks are based on the error rate of neuron networks output.

About CNN, we will find an appropriate activation function and the number of convolution layers and pooling layers.

Lastly, the fully-connected output layer needs a softmax activation to calculate the loss. To test how the model trains with respect to epochs, we can train the model on 20, 50, or more epochs.