ELEC 576 / COMP 576 – Fall 2021
Assignment 2

Due: Wednesday Oct. 27, 2PM CT via Canvas

Submission Instructions

Every student must submit their work in a zip file in the following format: netid-assignment2.zip. You should also provide intermediate and final results as well as any necessary code. No temporary files such as Tensorboard logs should be included in the zip.

1 Visualizing a CNN with CIFAR10

In this problem we will train a CNN on CIFAR10. After the network is trained, you will visualize the weights of the first convolutional layer to see what filters it learned. In addition, we will see how the activations look like by using the test images.

a) CIFAR10 Dataset: CIFAR10 is a dataset composed of natural images that represent 10 classes: *airplane*, *automobile*, *bird*, *cat*, *deer*, *dog*, *frog*, *horse*, *ship*, and *truck*. The dataset will be provided, and each image is grayscale and of size 28 x 28. Originally CIFAR10's images are RGB, and the original resolution is 32 x 32. We provided some additional preprocessing.

In Assignment 1, for MNIST it was easy to call a function to import the dataset. Here you will have perform this action. With `trainCifarStarterCode.py` and the zipped folder, `CIFAR10` , you will import all the images and perform additional preprocessing, such as dividing by 255. You should use one-hot encoding for labels.

b) Train LeNet5 on CIFAR10: In this part you will implement a LeNet5 and train it on CIFAR10. Here is the configuration of LeNet5:

- Convolutional layer with kernel 5 x 5 and 32 filter maps followed by ReLU
- Max Pooling layer subsampling by 2

- Convolutional layer with kernel 5 x 5 and 64 filter maps followed by ReLU
- Max Pooling layer subsampling by 2
- Fully Connected layer that has input 7*7*64 and output 1024
- Fully Connected layer that has input 1024 and output 10 (for the classes)
- Softmax layer (Softmax Regression + Softmax Nonlinearity)

Plot train/test accuracy and and train loss. In addition, try to search for good hyper-parameters (e.g. training methods, learning rate, momentum values ...). **Hint:** Run a few epochs for each set of hyper-parameters and see how train/test accuracy and train loss change.

c) Visualize the Trained Network: Visualize the first convolutional layer's weights. They should look like Gabor filters (edge detectors). Figure 1 shows examples of these filters. Notice that since you train your networks using gray-scale images, your filters will look slightly different. Also, show the statistics of the activations in the convolutional layers on test images.

2 Visualizing and Understanding Convolutional Networks

Read the paper Visualizing and Understanding Convolutional Networks by Matthew D. Zeiler and Rob Fergus. Summarize the key ideas of the paper.

(Optional) Visualization of features in a fully trained model: Apply one of the techniques discussed in the Visualizing and Understanding Convolutional Networks paper on the convnet trained in Problem 1. In the paper they used a validation data set, but in your case you would use the test set. Show your curiosity here :)

3 Build and Train an RNN on MNIST

An RNN is well suited for tasks on time-series or sequential data. However, in this problem we will see how we can use RNN for object recognition and compare it to Assignment 1 where we used a CNN.

a) Setup an RNN: With a CNN, we would send in full images, yet with the RNN, inputs to the network are 28 pixel chunks of the images. For example, each row of the image will be sent to the network at each iteration. The starter code, `lstmMNISTStarterCode.py`, provided will detail how it splits the data so that it can work for the RNN. You should modify the following parameters in the starter code.

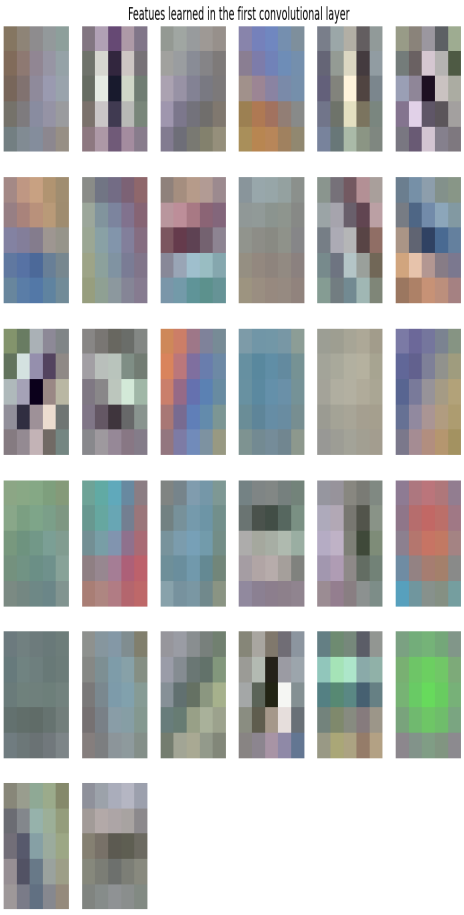

Figure 1: Filters learned in the first convolutiuonal layer

- Number of nodes in the hidden layer
- Learning rate
- Number of iterations
- Cost (hint use softmax cross entropy with logits)

- Optimizer

b) How about using an LSTM or GRU: change line 35 in the `starter code` (see below) to use LSTM and GRU instead of RNN.

```
rnn_cell.BasicRNNCell(nHidden)
```

Plot the train/test accuracies and the train loss. What do you notice and are the LSTM or GRU better? Also, change the number of hidden units and see how that affects the loss and accuracy.

c) Compare against the CNN: Compare with training using convnet in assignment 1 and describe any similarities or differences.

Collaboration Policy

Collaboration both inside and outside class is encouraged. You may talk to other students for general ideas and concepts, but each student is expected to work on their write-up independently.

Plagiarism

Plagiarism of any form will not be tolerated. You are expected to credit all sources explicitly.