ECE-GY 6143 / Spring 2020

Homework 5

Please scan and upload your assignments on or before April 16, 2020.

- You are encouraged to discuss ideas with each other; but
- you **must acknowledge** your collaborator, and
- you **must compose your own** writeup and/or code independently.
- We **strongly** encourage answers to theory questions in Latex, and answers to coding questions in Python (Jupyter notebooks).
- Please upload your solutions in the form of a single .pdf or .zip file on NYUClasses.
- Maximum score: 50 points.

---

1. **(10 points)** Consider a one-hidden layer neural network (without biases for simplicity) with sigmoid activations trained with squared-error loss. Draw the computational graph and derive the forward and backward passes used in backpropagation for this network.

   $$\hat{y} = W_2\sigma(W_1 x), \;\; \mathcal{L} = \|\hat{y} - y\|_2^2$$

   Qualitatively compare the computational complexity of the forward and backward passes. Which pass is more expensive and by roughly how much?

2. **(10 points)** Suppose that a convolutional layer of a neural network has an input tensor $X[i, j, k]$ and computes an output as follows:

   $$Z[i, j, m] = \sum_{k_1} \sum_{k_2} \sum_{n} W[k_1, k_2, n, m] X[i + k_1, j + k_2, n] + b[m]$$
   $$Y[i, j, m] = \text{ReLU}(Z[i, j, m])$$

   for some kernel W and bias b. Suppose X and W have shapes $(48, 64, 10)$ and $(3, 3, 10, 20)$ respectively.

   a. What are the shapes of Z and Y?

   b. What are the number of input and output channels?

   c. How many multiply- and add- operations are required to perform a forward pass through this layer? Rough calculations are OK.

   d. What are the total number of trainable parameters in this layer?

3. **(10 points)** The use of ℓ_2 regularization for training multi-layer neural networks has a special name: *weight decay*. Assume an arbitrary dataset $\{(x_i, y_i)\}_{i=1}^n$ and a loss function $\mathcal{L}(w)$ where w represents the trainable weights (and biases).

   a. Write down the ℓ_2 regularized loss, using a weighting parameter λ for the regularizer.

   b. Derive the gradient descent update rules for this loss.

c. Conclude that in each update, the weights are "shrunk" or "decayed" by a multiplicative factor before applying the descent update.

d. What does increasing λ achieve algorithmically, and how should the learning rate be chosen to make the updates stable?

4. **(20 points)** In this exercise, we will fine-tune a pre-trained deep network (ResNet34) for a particular two-class dataset which can be downloaded from the attached zip file. Code for pre-processing the dataset, and for loading ResNet34, can be found below. Since ResNet34 is for 1000 output classes, you will need to modify the last layer to reduce two classes. Train for 20 epochs and plot train and validation loss curves. Report your final train and validation accuracies.

```
import torchvision
from torchvision import datasets, models, transforms
transforms = transforms.Compose([
        transforms.Resize(256),
        transforms.CenterCrop(224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406],
                             [0.229, 0.224, 0.225])
    ])
train_set = datasets.ImageFolder("data/train",transforms)
val_set   = datasets.ImageFolder("data/val",transforms)

model = models.resnet34(pretrained=True)
```