EE 240: Pattern Recognition and Machine Learning
Homework 3

Due date: May 17, 2018

Description: Neural networks and deep learning.

Reading assignment and references: UFLDL notes, and Michael Nielson book

Homework and lab assignment submission policy:
All homework and lab assignments must be submitted online via https://iLearn.ucr.edu.
Homework solutions should be written and submitted individually, but discussions among students are encouraged.
All assignments should be submitted by the due date. There will be 25% penalty per day for late assignments. No grade will be given to homework submitted 3 days after the due date.

H3.1 Least-squares solution via gradient computation:

(a) Linear system: Let us consider a linear model $\mathbf{y} = W\mathbf{x} + b$, where W is an $m \times d$ matrix and $\mathbf{b} \in \mathbb{R}^m$. Write an expression for the optimal values of $W, \mathbf{b}$ that minimize the following cost function:

$$J(W, \mathbf{b}) = \frac{1}{2}\|\mathbf{y} - W\mathbf{x} - \mathbf{b}\|^2.$$

Remember the optimality condition that the gradient of the cost function w.r.t. the optimization variables should be zero at their optimal values. Therefore, first derive expressions for $\frac{\partial J}{\partial W_{ij}}, \frac{\partial J}{\partial \mathbf{b}_i}$, where W_{ij} denotes i, j entry in matrix W. Then set the gradients to zero to derive the expressions for the optimal solution. **(0 pts)**

(b) Convolutional system: Let us consider the following linear model $\mathbf{y} = W\mathbf{x} + \mathbf{b}$, where W is a Toeplitz matrix corresponding to a filter $\mathbf{w}$ of length k. We can write the $d + k - 1 \times d$ Toeplitz matrix as **(20 pts)**

$$W = \begin{bmatrix} \mathbf{w}_1 & 0 & 0 & \dots & 0 & 0 \\ \mathbf{w}_2 & \mathbf{w}_1 & 0 & \dots & 0 & 0 \\ \mathbf{w}_3 & \mathbf{w}_2 & \mathbf{w}_1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{w}_k & \mathbf{w}_{k-1} & \mathbf{w}_{k-2} & \dots & \dots & \dots \\ 0 & \mathbf{w}_k & \mathbf{w}_{k-1} & \dots & \dots & \dots \\ 0 & 0 & \mathbf{w}_k & \ddots & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \mathbf{w}_k & \mathbf{w}_{k-1} \\ 0 & 0 & 0 & \dots & 0 & \mathbf{w}_k \end{bmatrix}.$$

We can also define the linear model above as $\mathbf{y} = \mathbf{w} * \mathbf{x} + \mathbf{b}$, where $\mathbf{x} * \mathbf{w}$ denotes of two vectors that can be defined as

$$(\mathbf{x} * \mathbf{w})_j = \sum_{i=1}^{d} \mathbf{x}_i \mathbf{w}_{j-i+1},$$

where $\mathbf{w}_i$ denotes ith entry in vector $\mathbf{w}$ and $\mathbf{w}_i = 0$ for $i < 1$ and $i > k$.

Derive an expression for the optimal values of $\mathbf{w}, \mathbf{b}$ that minimize the following cost function:

$$J(W, \mathbf{b}) = \frac{1}{2}\|\mathbf{y} - \mathbf{w} * \mathbf{x} - \mathbf{b}\|^2.$$

Also, derive expressions for $\frac{\partial J}{\partial \mathbf{w}_i}$ and $\frac{\partial J}{\partial \mathbf{b}_i}$.

H3.2 In this question, we will learn how to implement backpropagation (or backprop) for "vanilla" neural networks (or Multi-Layer Perceptrons) and ConvNets. You will begin by writing the forward and backward passes for different types of layers (including convolution and pooling), and then go on to train a shallow ConvNet on the CIFAR-10 dataset in Python.

You can find detailed instructions on the starter code and data at the following link:
https://computing.ece.vt.edu/~f15ece6504/homework1/

(a) Two-layer Neural Network **(20 pts)**

The IPython notebook `two_layer_net.ipynb` will walk you through implementing a two-layer neural network on CIFAR-10. You will write a hard-coded 2-layer neural network, implement its backward pass, and tune its hyperparameters.

(b) Modular Neural Network **(40 pts)**

The IPython notebook `layers.ipynb` will walk you through a modular neural network implementation. You will implement the forward and backward passes of many different layer types, including convolution and pooling layers.

(c) ConvNet on CIFAR-10 **(20 pts)**

The IPython notebook `convnet.ipynb` will walk you through the process of training a (shallow) convolutional neural network on CIFAR-10.

Deliverables:

Zip the completed ipython notebooks and relevant files.

```
cd 1_cs231n
./collectSubmission.sh
```

Submit the generated zip file `1_cs231n.zip`.

Maximum points: 100