CS556: HOMEWORK 3
due 03/21/2017 by 1pm

Please submit your homework as a **single compressed folder** (e.g., zip file) to the TEACH website:

`https://secure.engr.oregonstate.edu:8000/teach.php?type=want-auth`

HW3 consists of two parts. The first part is about classifying images using a deep neural network, and the second part considers a mathematical derivation of a backpropagation algorithm.

Part 1 – 35 points

We will use the CIFAR-10 dataset that consists of 60000 32x32 images in 10 classes, with 6000 images per class. The class of an image is defined by the most prominent object in the image. The classes of CIFAR-10 images include: airplane, car, bird, cat, deer, dog, frog, horse, ship, and truck. There are 50000 training images and 10000 test images. The training and test images are randomly sampled from every class. CIFAR-10 images are provided on the class website in the MATLAB .mat file. You may also download the images from `http://www.cs.utoronto.ca/%7Ekriz/cifar.html`.

For classifying, CIFAR-10 images, we will develop two networks. Each network will consist of a number of convolutional layers and the soft-max layer. Training of the networks should be done in two stages: 1) Pre-training, and 2) Error Backpropagation. In pre-training, estimate parameters of each convolutional layer by minimizing the mean-squared error of representation, as illustrated in Fig. 1. Use 400 iterations over all training samples (i.e., 400 epochs) to pre-train each convolutional layer on the input features produced by the previous convolutional layer. Use 100 epochs to pre-train the soft-max layer on the input features produced by the last convolutional layer. After pre-training, run error backpropagation on the entire network in order to fine-tune all network parameters. After backpropagation, use the two networks to classify test CIFAR-10 images, and produce the corresponding confusion tables.

As illustrated in Fig. 2, Network 1 consists of two convolutional layers and the soft-max layer. The number of hidden units in the convolution layers is 200 and 100, respectively. Network 2 consists of three convolutional layers and the soft-max layer. The number of hidden units in the convolution layers is 200, 100, and 50, respectively.

Please submit to the TEACH website:

1) (10 points) Confusion table on CIFAR-10 test images for Network 1;
2) (10 points) Confusion table on CIFAR-10 test images for Network 2;
3) (5 points) All model parameters that you learned in fine-tuning for every computational layer of Network 1;
4) (5 points) All model parameters that you learned in fine-tuning for every computational layer of Network 2;
5) (5 points) Code that you used for training and testing Network 1 and Network 2.

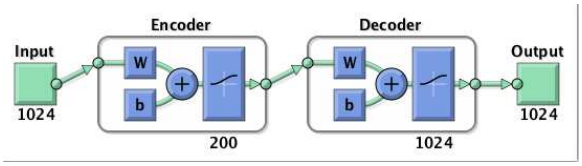

Fig. 1. Pre-training of a convolutional layer as an autoencoder.

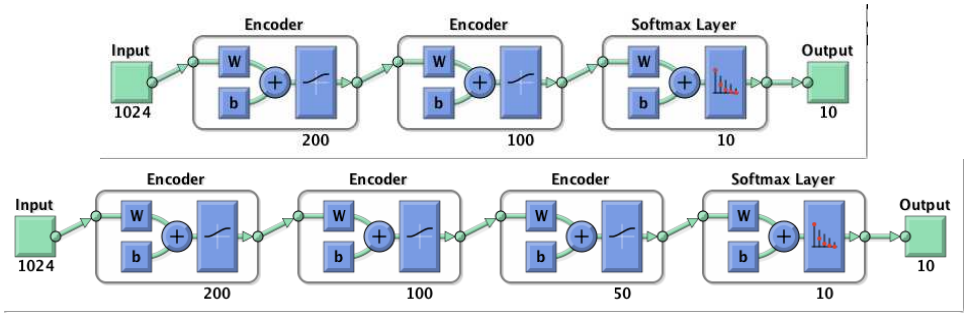

Fig. 2. (Top) Network 1; (Bottom) Network 2

Part 2 – 65 points

Derive a backpropagation algorithm for updating parameters w^l_{ji} of a deep neural network using the Newton Method:

$$w^l_{ji}(\tau+1) = w^l_{ji}(\tau) - \eta \left.\frac{E'}{E''}\right|_{w^l_{ji}(\tau)} \tag{1}$$

where

$$E = \frac{1}{2}\sum_k (t_k - a^L_k)^2 \ , \quad E' = \frac{\partial E}{\partial w^l_{ji}} \ , \quad E'' = \frac{\partial^2 E}{\partial {w^l_{ji}}^2}, \tag{2}$$

and t_k is the ground-truth (target) value for the neural activation:

$$a^L_k = f(\sum_j w^L_{kj} a^{L-1}_j + b^L_k) \ , \quad f(z) = (1+\exp(-z))^{-1}. \tag{3}$$

You should consider parameters of the top (output) layer, $l = L$, and the layer next to the top one, $l = L-1$, and use these derivations for generalizing to parameters of any layer in the deep neural network $l = 1, ..., L$.

Hint: Following Lecture 10, slides 23–37, we can derive:

$$\begin{aligned}
\frac{\partial^2 E}{\partial {w^L_{kj}}^2} &= \frac{\partial \frac{\partial E}{\partial w^L_{kj}}}{\partial w^L_{kj}} = \frac{\partial \frac{\partial E}{\partial a^L_k}\frac{\partial a^L_k}{\partial \text{net}^L_k}\frac{\partial \text{net}^L_k}{\partial w^L_{kj}}}{\partial w^L_{kj}}, && (4)\\
&= \left(\frac{\partial \frac{\partial E}{\partial a^L_k}}{\partial w^L_{kj}}\frac{\partial a^L_k}{\partial \text{net}^L_k} + \frac{\partial E}{\partial a^L_k}\frac{\partial \frac{\partial a^L_k}{\partial \text{net}^L_k}}{\partial w^L_{kj}}\right)\frac{\partial \text{net}^L_k}{\partial w^L_{kj}}, && (5)\\
&= \left(\frac{\partial^2 E}{\partial {a^L_k}^2}\frac{\partial a^L_k}{\partial \text{net}^L_k}\frac{\partial \text{net}^L_k}{\partial w^L_{kj}}\frac{\partial a^L_k}{\partial \text{net}^L_k} + \frac{\partial E}{\partial a^L_k}\frac{\partial^2 a^L_k}{\partial {\text{net}^L_k}^2}\frac{\partial \text{net}^L_k}{\partial w^L_{kj}}\right)\frac{\partial \text{net}^L_k}{\partial w^L_{kj}}, && (6)\\
&= \left(\frac{\partial^2 E}{\partial {a^L_k}^2}\left(\frac{\partial a^L_k}{\partial \text{net}^L_k}\right)^2 + \frac{\partial E}{\partial a^L_k}\frac{\partial^2 a^L_k}{\partial {\text{net}^L_k}^2}\right)\left(\frac{\partial \text{net}^L_k}{\partial w^L_{kj}}\right)^2, && (7)
\end{aligned}$$