E9 205 – Machine Learning for Signal Processing

Homework # 4
Due date: Nov.4, 2019

Coding assignment submitted to mlsp19 doT iisc aT gmail doT com

1. **Weight sharing in back propagation** - Modify the recursive backpropagation derivation (given in class) for the case when $\boldsymbol{W}^l = \boldsymbol{W}^{l+1}$ for some choice of l (where l denotes the layer l of the neural network).

   (**Points** 15)

2. **Neural Networks - Cost Function** - Let us define a NN with softmax output layer and $\{\mathbf{o}_i\}_{i=1}^M$ and $\{\mathbf{y}_i\}_{i=1}^M$ denote the input and targets to the NN. The task is classification with hard targets $\mathbf{y} \in \mathcal{B}^{C\times 1}$, where $\mathcal{B}$ denotes boolean variable (0 *or* 1), and C is the number of classes. Note that every data point $\mathbf{o}_i$ is associated with only one class label c_i where $c_i \in \{1, 2, .., C\}$ classes. The output of the network is denoted as $\mathbf{v}_i$ where $\mathbf{v}_i = \{v_i^1, v_i^2, .., v_i^C\} \in \mathcal{R}^{C\times 1}$ and $0 < \mathbf{v}_i < 1$. The NN cost function can defined using mean square error (MSE).

   $$J_{MSE} = \sum_{i=1}^{M} ||\mathbf{v}_i - \mathbf{y}_i||^2$$

   Show that the MSE is bounded in the following manner

   $$\sum_{i=1}^{M} \left[1 - v_i^{c_i}\right]^2 \leq J_{MSE} \leq 2 \sum_{i=1}^{M} \left[1 - v_i^{c_i}\right]^2$$

   (**Points** 15)

3. Azhar is doing his MLSP course assignment where he is asked to implement the back-propagation algorithm. He chooses a single hidden layer feedforward neural network (NN) with tanh activation function.

   $$g(a) = tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$$

   Given a set of N training samples $\{\mathbf{x_n}, \mathbf{t_n}\}_{\mathbf{n=1}}^{\mathbf{N}}$, he also uses linear output layer with a mean square error cost function defined as $E = \sum_n E^n$ where $E^n = \sum_n ||\mathbf{y_n} - \mathbf{t_n}||^{\mathbf{2}}$ and $\mathbf{y_n}$ is the output of the NN. He initializes his network with all the weight matrices (both W^1 and W^2) with $\mathbf{0}$. When he implements the backpropagation learning in python, he realizes that the weights do not get updated and stay at $\mathbf{0}$. Does he have a bug in his code ? (**Points** 20)

4. **Implementing BackPropagation**
   leap.ee.iisc.ac.in/sriram/teaching/MLSP_19/assignments/HW3/Data.tar.gz
   The above dataset has training/test subject faces with happy/sad emotion are provided in the data. Each image is of 100×100 matrix. Perform PCA to reduce the dimension from 10000 to $K = 12$. Implement a 2 hidden layer deep neural network (each layer containing 10 neurons) to classify the happy/sad classes. Use the cross entropy error function with softmax output activations and ReLU hidden layer activations. Perform 20 iterations of back propagation and plot the error on training data as a function of the iteration. What is the test accuracy for this case and how does it change if the number of hiddent neurons is increased to 15. Does momentum (with a momentum factor of 0.9) improve the learning process ? Provide your detailed analysis. (**Points** 30)

5. **M**NIST data - Download the dataset of hand-written digits
   http : //yann.lecun.com/exdb/mnist/
   containing 10 classes. [Reduce the number of samples in training data if your computing powers are limited as required by random subsampling]. The Keras package is needed for the rest of the question *https : //pypi.python.org/pypi/Keras*
   Use the Keras package to implement a DNN model with 2 and 3 hidden layers. Each hidden layer contains 512 neurons. What is the performance on the test dataset for this classifier. (**Points** 20)