COMP 421 – HOMEWORK 03

REPORT

Özgür Taylan Özcan

First, I read the training and test data from given data file, and placed the data in relevant matrices in which rows represent each image and columns represent pixel indices. (This part is the same with the Hw01 and 02)

Then I implemented the needed functions (sigmoid, softmax and safelog). I also set the learning parameters as stated in the homework description.

Next, I set the X (training data points) and R (truth labels for training data) matrices for training data, and initialized W & v with uniform random values (between -0.01 and 0.01).

I implemented a loop to apply the backproragation algorithm for training the multilayer perceptron. (Figure 11.11 from the book)

Initialize all v_{ih} and w_{hj} to rand$(-0.01, 0.01)$
Repeat
    For all $(\boldsymbol{x}^t, \boldsymbol{r}^t) \in \mathcal{X}$ in random order
        For $h = 1, \ldots, H$
            $z_h \leftarrow \text{sigmoid}(\boldsymbol{w}_h^T \boldsymbol{x}^t)$
        For $i = 1, \ldots, K$
            $y_i = \boldsymbol{v}_i^T \boldsymbol{z}$
        For $i = 1, \ldots, K$
            $\Delta \boldsymbol{v}_i = \eta (r_i^t - y_i^t) \boldsymbol{z}$
        For $h = 1, \ldots, H$
            $\Delta \boldsymbol{w}_h = \eta (\sum_i (r_i^t - y_i^t) v_{ih}) z_h (1 - z_h) \boldsymbol{x}^t$
        For $i = 1, \ldots, K$
            $\boldsymbol{v}_i \leftarrow \boldsymbol{v}_i + \Delta \boldsymbol{v}_i$
        For $h = 1, \ldots, H$
            $\boldsymbol{w}_h \leftarrow \boldsymbol{w}_h + \Delta \boldsymbol{w}_h$
Until convergence

In the loop, I used the sigmoid function to compute Z (hidden layers) values and softmax function to predict Y values for training data. And, I used gradient descents to learn W and v. (11.28 and 11.29 from the book)

$$\Delta v_{ih} = \eta \sum_t (r_i^t - y_i^t) z_h^t$$

$$\Delta w_{hj} = \eta \sum_t \left[\sum_i (r_i^t - y_i^t) v_{ih} \right] z_h^t (1 - z_h^t) x_j^t$$

I also captured the objective values for each iteration. The loop iterates until convergence or the maximum iteration is reached.

After backpropagation, I used a plot function to draw the objective function values captured through the iterations of the loop.

To form the confusion matrix for training data, I first created a matrix representing the truth values for each data point. Then created the confusion table using the predicted and truth values for training data.

To get the predicted Y values for test data, I used the following formula: Y <- X * W * t(v). After this, just like the training data, I formed a confusion matrix for the test data. Then I printed the two confusion matrices.