

CS59000: Machine Learning for Natural Language Processing

HOMEWORK 1

Due: Oct 5, 2017 on Thursday

1 Task

We are required to do the semantic analysis using perceptron and MLP. The inputs are simply the sentences and their corresponding labels. I make a bag of words from the training sentences and select top 3000 words. Also, I remove the **stop words** and reduce the words to its base word using **nlTK steammer**

2 Perceptron

2.1 The algorithm

Perceptron algorithm is used for supervised learning on binary data and is used as a linear classifier. The weights are updated using the gradient descent method. The basic algorithm is as follows:

1. Let's say we have input matrix \mathbf{X} which is of the dimension $[n \times p]$, where n are the number of examples and p are the number of features, input weight matrix \mathbf{W} of the dimension $[p \times 1]$, i.e. weight corresponding to each feature, bias \mathbf{b} and \mathbf{y} as our original labels
2. So the prediction, y_p is $\text{step_function}(W^T.X + b)$
3. $\text{Error} = y - y_p$
4. The weight is updated as $W = W + X^T * \text{Error} * lr$ and bias is updated as $b = b + \text{sum}(\text{Error}) * lr$ where lr is the learning rate
5. This updation takes for the defined number of epochs

2.2 Parameters

I found that the best parameter of my perceptron are as follows:

1. Learning rate = 0.09, Epochs = 1000

3 MLP

3.1 The algorithm

I have made a 3 layer neural network, Input layer, Hidden layer and an Output layer. MLP consists of many perceptron in the hidden unit with an activation function. I have used the sigmoid activation. The output unit over here is also a perceptron with sigmoid activation. The weights are trained using the back-propagation algorithm using gradient descent. It follows the basic concept of chain rule : find the error at the output layer, find the $\frac{\partial E}{\partial W_o}$ (W_o are the output layer weights), then with these two information, find the Error at the hidden layer and the same chain rule keeps going on for all the hidden layer.

Following are the terminologies associated to the 3 layer **MLP**

1. W_{out} : output layer weights, W_h : hidden layer weights, b_h : hidden layer bias, b_{out} : output layer bias.
2. Once we have the error E at a particular layer, we have the $\frac{\partial E}{\partial W}$ for that layer : $E * \text{sigmoid_gradient}(\text{activation})$ where activation is simple $\text{sigmoid}(\text{input} \times W + b)$. The the weight can be simply updates as : $W = W + \text{input}^T \times \frac{\partial E}{\partial W}$
3. The error back propogated from this layer to previous layer is : $\frac{\partial E}{\partial W} \times W^T$ where W is this layer weights which is the same as weights connecting previous layer to this layer.
4. The same chain rule keeps going on till we have covered all the hidden layers.
5. To avoid over fitting , there is also an another term regularization λ that penalizes large weights

3.2 Parameters

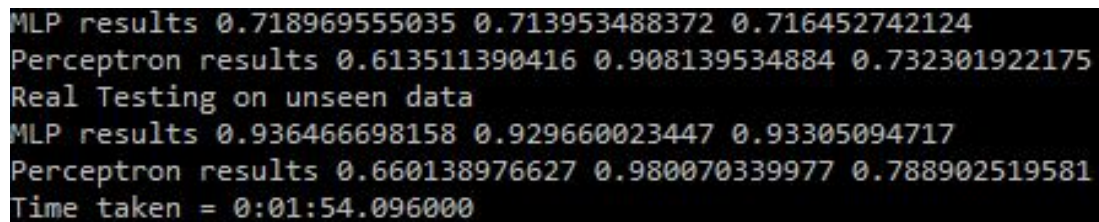
1. Learning rate : 0.01, number of hidden layer neurons = 10, epochs = 1000, $\lambda = 0.01$

4 Results

Results on one of the iteration by training on 80% and testing on the remaining 20% of the data are as follows:

Table 1: Precision, Recall , F score

	Perceptron	MLP
Precision	0.7046	0.732
Recall	0.8118	0.7154
F1 score	0.7544	0.7238



```
MLP results 0.718969555035 0.713953488372 0.716452742124
Perceptron results 0.613511390416 0.908139534884 0.732301922175
Real Testing on unseen data
MLP results 0.936466698158 0.929660023447 0.93305094717
Perceptron results 0.660138976627 0.980070339977 0.788902519581
Time taken = 0:01:54.096000
```

Figure 1: The output of the program on the terminal