MLP and Logistic Regression with TF

Micael Mota

I. Problem description

Given a digit image data-set, build and train a neural network using Multilayer Perceptron (MLP - with only one hidden layer) and using Logistic Regression by Gradient Decent with TensorFlow. The images dimensions are 71x77 and we can't process it before apply it in the net and input and output dimensions must be equal for both networks. We can use some optimization methods.

II. About the nets

Input dimension is (71x77) and output dimension is (1x10). The MLP hidden layer has 128 neurons and uses relu activation function, the output activation uses sigmoid.

III. Training

The data-set was randomly divided in 90% for train and 10% for accuracy validation. Both networks were trained for 50 epochs using Stochastic Gradient Descent (SGD) and learning rate decay optimization methods.
Mini batches sizes:
- Logistic Regression = 32
- MLP = 16

A. Training charts

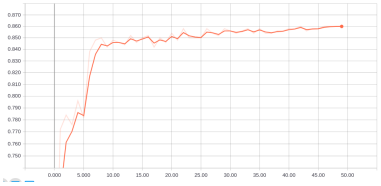

Fig. 1. Logistic Regression Accuracy

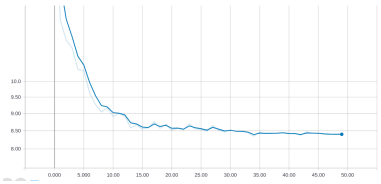

Fig. 2. Logistic Regression Loss

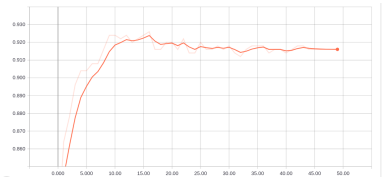

Fig. 3. MLP Accuracy

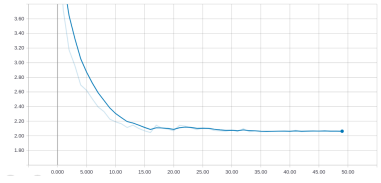

Fig. 4. MLP Loss

IV. Results

Train:

Method	Loss	Accuracy
Logistic Regression	5.030	0.916
Multilayer Perceptron	0.413	0.981

Validation:

Method	Loss	Accuracy
Logistic Regression	7.923	0.858
Multilayer Perceptron	1.904	0.928

How to use

Run 'python3 tf-lr.py' for Logistic Regression or 'python3 tf-mlp.py' for MLP and type '1' for training or '2' for inference on the 'data_part1/test' images. The program will output a text file in 'outputs/tf' with one line for each image in test folder with respective prediction.