HY587: Assignment 2

Build, Train & Test a Multilayer Neural Networks using Tensorflow

Issued: 24 April 2020
Deadline: 8 May 2020, 23:59

Description

The goal of this exercise is to implement, train and test a Multi-Layer Feed Forward Neural Network using the SGD algorithm and automatic differentiation based on the TensorFlow framework.

You will use the Digits dataset (10 class handwritten digits) for the task of image classification. The given source code automatically downloads and processes the dataset using the Scikit-learn Python library (if you use Anaconda distro, it is already installed in your Python environment).

You can use any available function of NumPy, SciPy, TensorFlow, TensorBoard required for the following tasks.

1. Build and train a feed-forward neural network using the Stochastic Gradient Descent method. Use the set of hyperparameters provided in the source code (lines $170 - 180$).

   - Build the structure of your model connecting the input layer of your net to one hidden layer h of size $hid\_size = 15$ and then to an output layer. Create the weights W_h and the bias b_h required for the hidden layer as well as those for the output layer W_o, b_o. Your output layer produces 10 values z_j^o, $j = \{0, .., 9\}$, matching the number of the target classes of the dataset.
   - Use the sigmoid as your activation function $h(x)$.
   - Use the Softmax cross entropy error function to estimate the probability errors between the scores of the output neurons o_j in your model and the target labels t_j of your training samples, where $t_j \in \{0, 1\}$ and $t_j = 1$ iff $j =$ correct class and $j = 0$ otherwise, for $j = \{0, .., 9\}$, for each sample of the training batch/set that contains N training samples.

     The scores z^o of your output neurons are computed as:

     $$z^o(x) = W^o h(x) + b^o, \tag{1}$$

     where $h(x)$ are the activation responses of the neurons in the hidden layer, W^o are the weights and b_o the bias.

     Let o_j denote the resulting softmax function for each output neuron j:

     $$o_j = softmax(z_j^o) = \frac{e^{z_j^o}}{\sum_j e^{z_j^o}}, \tag{2}$$

where the sum in the denominator is over each neuron in the output layer.

Then the cross entropy function $E(t;o)$ for a training sample $i \in \{0, N-1\}$ is defined as follows:

$$E(t,o) = -\sum_j t_j log(o_j), \tag{3}$$

- Finally, compute the loss of your model to be minimized as the sum of E for all samples x of your training batch/set.

$$loss = \sum_{i=1}^{N} E(t(i), o(i)), \tag{4}$$

2. Train your model and check the generalization on the test samples.
   - Train your model using the Stochastic Gradient Descent algorithm and mini-batches and check accuracy on the train set.
   - Compute predictions on the test set.
   - Compute average accuracy of the model the test set.
   - Visualize the graph of your model, the histories of loss and accuracy scores on train/test sets and other quantities (i.e. weights) you may find useful using TensorBoard.
3. In order to maximize the performance on the given dataset try different settings for your model:
   - A. Experiment with different hyperparameters (e.g. learning rate = 0.001,..,0.1, batch size = 8,..,128, size of hidden layers = 5,..,25, number of epochs).
   - B. Try different activation functions (e.g., ReLU, TanH).
   - C. Try to add more hidden layers (using same or different activation functions) and increase their size.
   - D. Add L2 regularization (e.g., with regularization strength 10^{-4})

BONUS: A +15% will be distributed to the top-performing models based on the accuracy on the test set (e.g if there are K submissions with equal top performance, each one will get a bonus $15\%/K$).

Notes

Setup your Anaconda, Python, TensorFlow environment

Installation guidelines for Anaconda, Python, TensorFlow, TensorBoard, Scikit-learn, (optional: Jupyter).

Python 3.5, TensorFlow 1.0 and Scikit-learn library are required to run the provided code and accomplish your assignment. Read the setup-README.txt file in the main folder of the assignment and the slides of the first tutorial to get info on how to setup your Python environment successfully. Link to slides of the course tutorials
`https://drive.google.com/open?id=1RanpIS8z72JlguTZAe1HsIfG7U4co61XrY_2nWoynOk`

Recommendation: Install Anaconda to get everything installed except TensorFlow+TensorBoard. Then follow the installation guide for TensorFlow to get done with the requirements. `https://www.tensorflow.org/install/`

You are also advised to follow the official TensorFlow and TensorBoard tutorials:
`https://www.tensorflow.org/tutorials/mnist/tf/`
`https://github.com/tensorflow/tensorflow`
`https://www.tensorflow.org/get_started/summaries_and_tensorboard`
`https://github.com/tensorflow/tensorflow/blob/master/tensorflow/tensorboard/README.md`

Dataset

You do not need to manually download the Digits dataset. It will automatically be set once you run the given source code. Check the provided comments in your code for more details on the Digits dataset.

Submission info

- Create a .pdf or .doc file to report the resulting scores, images/figures and any other comments or description of your work you may need to submit. Do not forget to include your name, login, ID in the report. Save this file in your working folder.
- Use zip/rar/gz to compress your working folder and rename it to cs587_my login_assignment2.xxx in order to submit a single file.
- You will use the following link of Dropbox request to upload your submission as a SINGLE zip/rar/gz file. You will be requested fill in your first-name/surname and an email address to upload. `https://www.dropbox.com/request/da4ekW9v4KmwTZrCbBJA`
- You do not need to have a Dropbox account to upload your submission.
- You can upload your submission as many times as you need keeping the same filename.
- Uploading will not be available after the deadline date-time.

Troubleshooting

In case you find any errors/bugs in the code please send an email to konarak@csd.uoc.gr or the mailing list.