CS-587: Assignment 2B

Build, Train & Test a Multilayer Neural Networks using TensorFlow

Issued: 05 April 2023
Deadline: 27 April 2023, 23:59

Description

The goal of this part of the assignment 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 distribution, 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 $15\%/K$).

Important Notes

Setup your Anaconda, Python, TensorFlow environment

For this assignment, you need to create a new working environment with Python 3.5, TensorFlow 1.15 framework and Scikit-learn library in order to run the provided code.

To create a new environment follow the guidelines mentioned in the first tutorial to setup your Python environment successfully. To install Tensorflow/TensorBoard follow the installation guidelines mentioned in the 2nd Class Tutorial. DO NOT install the tensorflow/tensorboard versions shown in the Anaconda GUI, since these refer to 2.X versions.

Link to slides of the course 2nd tutorial:
`https://drive.google.com/drive/folders/1mRtAlPuzaUw4rC_VQCUe7L1ZPX_C6VJK?usp=sharing`

Important: Due to major revisions between Tensorflow versions 1.X and the current 2.X, we will be sticking with Tensorflow 1.15 for our assignment. Please DO NOT rely on the documentation of the official Tensorflow page. Instead, you can find Tensorflow 1.15 and TensorBoard documentation in the following links:
`https://devdocs.io/tensorflow~1.15/`
`https://github.com/tensorflow/docs/tree/master/site/en/r1/guide`
`https://itnext.io/how-to-use-tensorboard-5d82f8654496`
`https://github.com/tensorflow/tensorflow/blob/master/tensorflow/tensorboard/README.md`

Also, you can search the web for answers regarding Tensorflow commands, BUT do not forget to specify the version we are using.

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 (preferred) 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 and 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_mylogin_assignment2b.xxx in order to submit a single file.
- The submission of your implementation will be via the e-learn platform. Submissions via e-mail will not be accepted.
- 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 kbacharidis@csd.uoc.gr or the mailing list.