EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

MATH. - NATURWISS. FAKULTÄT
FACHBEREICH INFORMATIK
KOGNITIVE SYSTEME · PROF. A. ZELL

Deep Neural Networks

Assignment 6

Assignment due by: 04.12.2019, Discussions on: 11.12.2019

Note: All further programming exercises in this course have to be done in Python 3.5 or later versions. To ensure that we can run your code, you are not allowed to use any external Python libraries except for NumPy, SciPy, Matplotlib and TensorFlow 2.0. Because we want you to understand what Deep Neural Networks in general and TensorFlow in particular are doing, **you are not allowed to use the high-level API Keras** (`tf.keras`) except for the functions already provided in the template.

This will be a programming-only assignment sheet. Make sure you only upload the two scripts as described in Question 1 and Question 2 together with the documentation for Question 2 (**as a PDF**) as a ZIP-file to Ilias. **Do not upload Jupyter notebooks files (.ipynb files).**

This week's assignment will serve as an introduction to TensorFlow 2.0, which is a framework for deep learning developed by Google. As you have probably seen during the last two weeks, computing gradients by hand is tedious. We chose TensorFlow because it is very versatile and has its high-level interface primarily in Python, which we have used so far in this course.

In this course, we want to use a low level interface of TensorFlow because it makes it easier to see the mechanics behind the networks. With this knowledge it should be easier to pick up the higher level interfaces TensorFlow provides.

Question 1 Getting started with TensorFlow (10 points)

Your first task is to set up TensorFlow 2 on your own pc/laptop. You can find the instructions on how to do this at `https://www.tensorflow.org/install`; we recommend that you install the CPU-only version, since setting up CUDA can take much more time and can have many problems.

The low level API tutorial (`https://www.tensorflow.org/programmers_guide/low_level_intro`) gives an overview on how to construct and run a computation and simple models. Also read through the two subsequent tutorials on variables and tensors. We have prepared a file (`tf_hands_on.py`) with some small exercises to get you used to TensorFlow's conventions. For each section, you should create the necessary operations to have TensorFlow perform the desired computation and then test that it is performed correctly. Complete the file `tf_hands_on.py` and upload this filled file together with the file from Question 2 and the PDF documentation in a ZIP-file to Ilias. Do not rename this file and make sure your names are included in the file.

Question 2 A simple neural network in tensorflow (4+2+2+2 points)

We have prepared a file (`simple_network.py`) which uses TensorFlow to implement a simple linear model for classifying the MNIST dataset which we already dealt with in Question 1 of Assignment 4. Please make sure you understand what the code is doing. The accuracy of this simple linear model is not very good, so we will try to improve it in this question by modifying the code.

(a) Introduce a hidden layer into the neural network. This means that we create a second set of weights and biases and our network should compute the function

$$\boldsymbol{S}(f(\boldsymbol{x}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2)$$

instead of

$$\boldsymbol{S}(\boldsymbol{x}\boldsymbol{W} + \boldsymbol{b})$$

which the linear model used ($\boldsymbol{S}$ is the softmax function). In addition to the extra bias vector and weight matrix, the new classification function contains an activation function f and is there to prevent the model from degenerating to a linear model. A common choice for f is the rectified linear unit (ReLU, $f(x) = \max(0, x)$, `tf.nn.relu()`).
We have to make a choice about how large the hidden layer should be (the size of $\boldsymbol{b}_1$ and the second/first dimension of $\boldsymbol{W}_{1/2}$) since it is not determined by the input or output size. For now choose the size to be 50. After you have included this hidden layer, make sure you provide a plot in the PDF-documentation of the evolution of the train and validation loss and accuracy.

(b) Change the size of the hidden layer to 20 and 70, respectively, and train the model. Again provide plots in the PDF documentation.

(c) Introduce a second hidden layer (with bias and ReLU) into the model and see how high you can get the accuracy while iterating over 100 epochs at most (size of the layers does not matter). Write down which size of the hidden layer yielded the best result and again provide plots in the PDF documentation.

(d) Now see how accurate you can make the model with the restriction that the total number of neurons in all hidden layers (potentially more than 2) can be no more than 12000. Try and document at least 2 different configurations and provide plots. For this last part you are also allowed to vary the step size and activation function.