Neural Network and Deep Learning

Assignment 2

Due Monday, 17 October, 11:59 pm.

Modify Chollet’s code from Section 2.1 of his book, 2.1\_Mnist\_Chollet.ipynb.

The notebook is well documented, explaining the details of the code. Also we’ve mentioned the MNIST hand written digits classification in lecture.

In many places in the world, you can just run this code without much problem. Chollet uses one line of Keras code to download the 60000 training set and 10000 testing set into the notebook.

(train\_images, train\_labels), (test\_images, test\_labels) = mnist.load\_data()

However in China, we are located far from the official storage location for this dataset and other Tensorflow and Keras tools. So we have to make some changes in order to run the code.

Last assignment, you were introduced to the Tsinghua University mirror site. This assignment, we download the dataset off the Internet and manually load the dataset into the notebook.

You have to be familiar with these techniques to be a successful computer scientist or programmer in China.

**Modify Chollet’s 2.1 code:**

(1) Have your code print your name and ID number at the first cell.

(2) 2.1\_Mnist\_pages1\_2.pdf contains the load\_data() function. Incorporate this function into your code and use this function to load the data.

Note the pdf also contains other little changes, such as

#import keras

import tensorflow.keras as keras

If you don’t make these changes, the code will give you a warning when you run the code. It is a good practice to make these changes.

(3) Download mnist\_dataset in iSpace. Extract this dataset, and place it next to your Jupyter Notebook code.

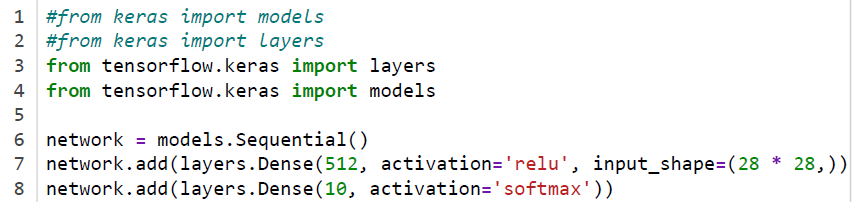
(4) Run the code. When you finished debugging the code, do Restart and Run All to get the cell numbers in the proper order. Print the code as a pdf. Rename the code as 2.1(YourName)(YourID).pdf; of course, put in your own name and ID number. Submit the pdf file in iSpace.

---------------------------------------------------------------------------------------

Don’t just run the code, read the explanations in the jupyter notebook, so you understands what is going on.

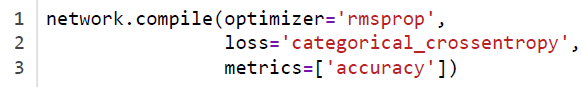
|  |  |
| --- | --- |
|  | Training data:   * 60,000 training images, each is 28 x 28 pixels * 60,000 training labels * Image 0 is a 5, image 1 is a 0, image 2 is a 4, … |

|  |  |
| --- | --- |
|  | Testing data:   * 10,000 testing images, each is 28x28 pixels * 10,000 testing labels * Image 0 is a 7, image 1 is a 2, image 3 is a 1, … |



Line 7: first layer is a densely connected layer with 512 nodes, use ReLU activation function, input image has 28 x 28 pixels

Line 8: second layer is a densely connected layer with 10 nodes, with softmax activation function; this classifies the image into 10 classes, corresponding to the digits 0 to 9.



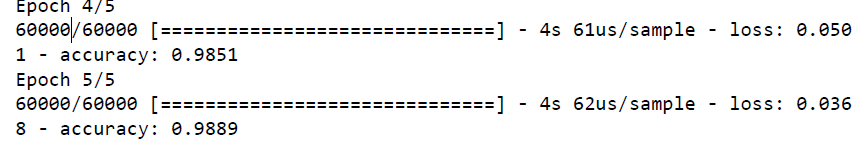
Use rmsprop, an alternative to gradient descent; will learn more in C2M2L07.

Use the categorical crossentropy loss function (C1M2L03)

Output will display accuracy = 100% - error rate; (will learn other possibilities later)

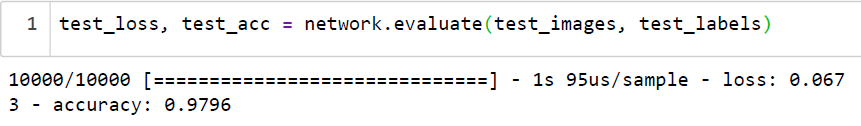


Fit the model to the training data, using 5 epochs iterations (similar to the iteration steps we mentioned in gradient descent), mini-batch size = 128 (randomly select 128 images at each mini-step, C2M2L01).



At the 5th epoch (step), a training accuracy of 98.89% is reached.

After the model is trained, run the test data:



After running all 10,000 test images, we achieved 97.96% accuracy.