Colab, Loss Function, Neural Network

EECE-4710 IoT and Machine Learning

Cristinel Ababei
Electrical and Computer Engr., Marquette University

1. Objective

To practice more Python coding using Colab, which is introduced here for the first time. We investigate the loss cost function for a simple regression model. A simple neural network is constructed and studied using TensorFlow and Keras. Also, callbacks in Keras are illustrated to study the impact of epochs on model performance during model training.

2. Colab

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with: zero configuration required, access to GPUs free of charge, and easy sharing. One can run Jupyter Notebooks, which is very nice. *Jupyter Notebook* is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Here is an introduction to Google Colab - a 3 minutes video with Jake VanderPlas from Google giving a great intro to Colab:

- https://www.youtube.com/watch?v=inN8seMm7UI

And here are some tips to remember when using Colab:

- https://raw.githubusercontent.com/Mjrovai/UNIFEI-IESTI01-TinyML-2022.1/main/00_Curse_Folder/1_Fundamentals/Class_04a/docs/Tips_for_using_Colab.pdf

3. TensorFlow and Keras

TensorFlow is the second machine learning framework that Google created and used to design, build, and train deep learning models. TensorFlow provides APIs for Python, C++, Haskell, Java, Go, Rust, and there's also a third-party package for R called tensorflow.

It is a library to do numerical computations - but these computations are done with **data flow graphs**. In these graphs, **nodes represent mathematical operations, while the edges represent the data**, which usually are **multidimensional data arrays or tensors**, that are communicated between these edges. So, one could say that the name "TensorFlow" is derived from the operations which neural networks perform on multidimensional data arrays or tensors; it is literally a flow of tensors.

Python Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK or Theano. Being able to go from idea to result with the least possible delay is key to doing good research. **So, Keras was developed with the main focus on enabling fast experimentation.**

4. Refreshing Python (Optional)

Example 1: If your Python programming skills are rusty, then, please work through the following Notebook (included in the files and materials of this week provided in this course), which will give you the opportunity to get used to using Colab as well. This will be done in part during the lecture too.

- Python_Refresh_Notebook.ipynb

Which is available on GitHub at:

- https://github.com/Mjrovai/UNIFEI-IESTI01-TinyML-2022.1/blob/c37a9d83624bd425e889462a78985472b87f542c/00_Curse_Folder/1_Fundamentals/Class_04a/notebooks/Class_4a_Python_Overview.ipynb

To do that, open Colab:

- https://colab.research.google.com/

Then, click on File->Upload Notebook and select from your computer:
Python_Refresh_Notebook.ipynb
Or, click on File->Open Notebook and select GitHub and paste the above GitHub link, then, click on it once found. This will open the notebook, which you should go through step by step (after you first do Edit->Clear All Outputs).

5. Exploring Loss (Cost) Function Variation

Example 2: In this exercise we do step-by-step the Notebook: Exploring_Loss_Cost_Function.ipynb
To do that start Colab, then, click on File->Upload Notebook and select from your computer:
Exploring_Loss_Cost_Function.ipynb
Once you uploaded the Notebook, do first Edit->Clear All Outputs.
The exercise looks at how the loss (cost) function changes with w and b when the following expression: **`Y=w*X+b`** is used to model the relationship between the following input data and output target:

```
X = [-1, 0, 1, 2, 3, 4]
Y = [-3, -1, 1, 3, 5, 7]
```

The loss function is calculated as the MSE (Mean Squared Error) between actual targets and estimations with the expression **`Y=w*X+b`** for different values of **w** and a fixed **b**.

6. First Neural Network in Colab with TensorFlow

Example 3: In this example we look at a very simple neural network with just 1 neuron!
To do that start Colab, then, click on File->Upload Notebook and select from your computer:
TF_First_Neural_Network.ipynb
Once you uploaded the Notebook, do first Edit->Clear All Outputs.
Then, work through it and observe everything – especially the last plot Loss vs. Epoch, which indicates that 100 epochs would have been enough to train this simple ML model.

Example 4: Next, we look at another Notebook, which explores EPOCHS using CallBacks:
TF_First_Neural_Network_v2_exploring_epochs.ipynb
Upload it in Colab and execute all cells step-by-step. In this example, Keras callbacks are used to study the impact of epochs on model performance during model training. Callbacks can be passed to keras

methods such as fit, evaluate, and predict in order to hook into the various stages of the model training and inference lifecycle. Read more about them here:
https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/Callback

7. Assignment

Part 1: Redo the example from **Exploring_Loss_Cost_Function.ipynb**, and change/expand W_lst and b_lst as shown below, then, create at the end two plots. Each plot would contain 12 graphs; for example, the first plot should have 12 individual graphs, each corresponding to an individual value of b, and plotted for the same W_list on x-axis.

```
W_lst = [-4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7]
plt.plot(W_lst, MSE_lst, color='orange', marker='o', label='error')
b_lst = [-5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6]
plt.plot(b_lst, MSE_lst, color='orange', marker='o', label='error')
```

Part 2: Modify the example **TF_First_Neural_Network.ipynb**
and change the simple NN model to increase the number of units on layer from 1 to: 2,3,4
Re-test the new models and compare to the initial model. You will have to include the results from testing (including the new Loss vs. Epoch plots in the report).

Next, vary the number of layers in the model and re-test. Start with the initial base model but with 4 units on one layer. Then, add a second, a third and a 4th layer, each with 4 units per layer. Each time, re-test the model and compare the results. To add layers to the model, you can do (see second file in Readings/):

```
my_layer_1 = keras.layers.Dense(units=2, input_shape=[1])
my_layer_2 = keras.layers.Dense(units=1)
model = tf.keras.Sequential([my_layer_1, my_layer_2])
```

In this part you should have created two sets of three plots. Each plot should show the reference or base case and the new one obtained with each of the new parameter values you need to change. For example, the first plot should show the graphs for the NN with one layer and with 1 (base) and 2 (new value) neurons.

8. Deliverables

You must write (typed) a report and upload it as a PDF file on D2L. The report should be named "**LastName_hw2.pdf**". The report should include the following sections:

1) Title + course info + your name
2) **Summary.** Describe in one paragraph what the objective of the assignment is.
3) **Exploring Loss Function.** Describe what the loss function is in the context of the Part 1 assignment. Include the new plots as figures (with captions!) and discuss those in several sentences.
4) **Neural Networks.** Describe your experiments. Include all plots that you created as well as the inference results you observed with: model.predict([10.0]). Discuss your results. Which model is the best?
5) **Conclusion.** Present your conclusions and describe what issues you encountered and how you solved them.
6) **References.** Include all references that you used, as a numbered list. Cite them in the report itself; do not just list them here.