Advanced Machine Learning T1 Question Paper

Scope of this exam:

Unit 1: TensorFlow and Keras

Objectives of this project

- Get a development experience on building custom estimators for TensorFlow
- Develop a broad based understanding of how to choose hyperparameters such as number of hidden units in a neural network

Problem Statement:

In this exam you are required to develop a custom estimator for TensorFlow 1.9 or later that implements the architecture described in the steps below. You are required to train, validate and test your implementation using the dataset provided for this exam.

Detailed Steps:

1. You are required to provide the same functionality as the premade estimator DNNRegressor that we covered during earlier lab sessions. The reference for this is: https://www.tensorflow.org/api_docs/python/tf/estimator/DNNRegressor
2. You need to add 2 features to this standard estimator:
    a. A method that returns the parameters of the layer, given an identifier for the layer. The identifier can be the layer name or its index in the stack of a deep network. For example: if we have a dense layer with name "dense1", we may get the parameters as: weights, biases = model.get_layer_params(id)
    b. A method that returns the activations of the layer given the layer name or index. For example: layer_activations = model.get_layer_activations(id).

    Note: In the above examples, model is the instance of the custom estimator that you will create.
3. Your implementation should support a vector as input and a vector as output. Volume tensors (e.g. arising out of images etc) or time series sequences (e.g. LSTM) are not required to be supported. Your layers are Dense layers supporting all the activation functions (such as tanh, relu) that can be vertically stacked.
4. Once you build the custom estimator (let us call it MyDNNRegressor), you should test it with the dataset provided for this exercise: algebra.csv
5. The dataset has 20000 rows and 4 columns. The first 2 columns of the CSV are features and the last 2 columns are the target outputs that we should train for.
6. Use 85% of this data for training and remaining for validation
7. Firstly, you should choose column 1 and column 2 as feature vector and choose column 3 as the target attribute. Use the feature_columns parameter of your MyDNNRegressor. This should be used along with Data API of TensorFlow
8. Build a network with MyDNNRegressor where you define an input, hidden and output

layer for dimensions 2, 2, 1.

9. After training, use the get_activation() method to get the activation of your hidden layer. This will have precisely 2 scalar numbers. Use the get_parameters() for the same layer and obtain weight and bias values. Using these compute: a1 * w1 + b1 and a2 * w2 + b2. Report these values. Determine the value of final output and looking at all these, guess the function: f(x1, x2)
10. Your custom estimator should provide all the functions that the standard Estimator of TensorFlow provides. For example, you should be able to specify a models folder and save your models.
11. Now, build another model as in Step 8, train the model for same features but choose the last column as the target
12. Repeat the above and report the values as specified in Step 9 above
13. Lastly, build a model where you define an input, hidden and output layer for dimensions 2, 2, 2
14. Measure the outputs, observe the behavior, report the findings
15. Change the number of hidden units to be 3 instead of 2, repeat and report the results
16. Write a report and place it in a zip folder along with your code, any other material like screenshots etc. The report should describe your results for each of the tasks as in steps 9, 12, 14, 15
17. Send the deliverables to: panantharama@alum.iisc.ac.in

Enjoy the session, best wishes from the faculty!

P.N. Anantharaman, Ambili Rajan