TensorFlow: Qwik Start

Building a system trained on data to infer the rules that determine a relationship between numbers.

Main Steps:

1. Prepare the data: prepare the data your model will be trained on.

2. Design the model:

   Create a neural network, it should have a layer and a neuron.

   Properties:

   - Architecture Definition: which model do you want to create? Ex: sequential model...

   - Layer Type and Parameters:
     - single dense layer or multiple layers?

     Ex: (‘tf.keras.layers.Dense’)

     - How many neurons do you want to create?

     Ex: (‘units=1’)

- Input Shape: Determine the shape of the input data that the model expects.

Ex: input_shape=[1] indicates that the model expects - - input data with one feature.

3. Compile the model:

- write the code to compile your neural network. When you do, you must specify 2 functions, a loss and an optimizer.

- Loss Function: The loss function measures the guessed answers against the known correct answers and measures how well or how badly it did.

Ex: tf.keras.losses.MeanSquaredError():

- Mean Squared Error (MSE) is a common loss function used for regression problems.
- It calculates the mean of the squared differences between the predicted values and the actual values.
- The goal during training is to minimize this value, as it indicates how far the model's predictions are from the actual targets.

- Optimizer Function: It makes another guess. Based on the loss function's result, it will try to minimize the loss.

Ex: tf.keras.optimizers.SGD() :

- Stochastic Gradient Descent (SGD) optimizer is used for optimizing the model's weights during training.
- It updates the model's parameters (weights) in the direction that minimizes the loss function.

4. Train the neural network:

- To train the neural network to 'learn' the relationship between the Xs and Ys, you will use model.fit.
- This function will train the model in a loop where it will make a guess, measure how good or bad it is (aka the loss), use the optimizer to make another guess, etc. It will repeat this process for the number of epochs you specify, which in this lab is 500.

Ex: model.fit(xs, ys, epochs=500)

- In the above code model.fit will train the model for a fixed number of epochs.

5. Run your script: python model.py
6. Using the model: cloud_logger.info(str(model.predict([10.0])))

Link:

https://www.cloudskillsboost.google/course_templates/646/labs/455572