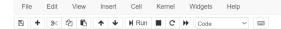
coursera



Jupyter C2\_W4\_Lab\_1\_basic-mirrored-strategy Last Checkpoint: 01/21/2021 (autosaved)



Trusted Python 3 O



```
Open in Colab
```

## Mirrored Strategy: Basic

In this ungraded lab, you'll go through some of the basics of applying Mirrored Strategy.

## **Imports**

Load the MNIST dataset and split it into training and test chunks.

Next, you define strategy using the MirroredStrategy() class. Print to see the number of devices available.

## Note:

- . If you are running this on Coursera, you'll see it gives a warning about no presence of GPU devices.
- . If you are running this in Colab, make sure you have selected your Runtime to be GPU for it to detect it.
- In both these cases, you'll see there's only 1 device that is available.
- One device is sufficient for helping you understand these distribution strategies.

WARNING:tensorflow:There are non-GPU devices in `tf.distribute.Strategy`, not using nccl allreduce.

 ${\tt WARNING: tensorflow: There \ are \ non-GPU \ devices \ in \ `tf. distribute. Strategy`, \ not \ using \ nccl \ all reduce.}$ 

 $INFO: tensor flow: Using \ Mirrored Strategy \ with \ devices \ ('/job:local host/replica: 0/task: 0/device: CPU: 0',)$ 

 $INFO: tensor flow: Using \ Mirrored Strategy \ with \ devices \ (''job:local host/replica: \theta/task: \theta/device: CPU: \theta',) \\$ 

Number of devices: 1

Next, you create your training and test examples, define your batch size and also define BATCH\_SIZE\_PER\_REPLICA which is the distribution you are making for each available device.

```
In [4]: W # Get the number of examples in the train and test sets
num_train_examples = info.splits['train'].num_examples
num_test_examples = info.splits['test'].num_examples

BUFFER_SIZE = 10000

BATCH_SIZE_PER_REPLICA = 64
# Use for Mirrored Strategy
BATCH_SIZE = BATCH_SIZE_PER_REPLICA * strategy.num_replicas_in_sync
# Use for No Strategy
# BATCH_SIZE = BATCH_SIZE_PER_REPLICA * 1
```

A mapping function which normalizes your images:

```
In [5]: When the for normalizing the image
def scale(image, label):
    image = tf.cast(image, tf.float32)
    image /= 255
    return image, label
```

Next, you create your training and evaluation datesets in the batch size you want by shuffling through your buffer size.

```
In [6]: # Set up the train and eval data set
train_dataset = mnist_train.map(scale).cache().shuffle(BUFFER_SIZE).batch(BATCH_SIZE)
eval_dataset = mnist_test.map(scale).batch(BATCH_SIZE)
```

For your model to follow the strategy, define your model within the strategy's scope.

- Run all the cells below and notice the results.
- Afterwards comment out with strategy.scope(): and run everything again, without the strategy. Then you can compare the results. The important
  thing to notice and compare is the time taken for each epoch to complete. As mentioned in the lecture, doing a mirrored strategy on a single device (which
  our lab environment has) might take longer to train because of the overhead in implementing the strategy. With that, the advantages of using this strategy
  is more evident if you will use it on multiple devices.

```
metrics=['accuracy'])
Fnoch 1/12
        WARNING: tensorflow: From /opt/conda/lib/python3.7/site-packages/tensorflow/python/data/ops/multi_device_iterator_ops.py:601:
get_next_as_optional (from tensorflow.python.data.ops.iterator_ops) is deprecated and will be removed in a future version.
        Instructions for updating:
        Use `tf.data.Iterator.get_next_as_optional()` instead.
        WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow/python/data/ops/multi_device_iterator_ops.py:601:
get_next_as_optional (from tensorflow.python.data.ops.iterator_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use `tf.data.Iterator.get_next_as_optional()` instead.
        Epoch 2/12
        938/938 [=========] - 25s 26ms/step - loss: 0.0372 - accuracy: 0.9889 Epoch 5/12 938/938 [===========] - 25s 26ms/step - loss: 0.0296 - accuracy: 0.9910
        Epoch 6/12
        938/938 [=============] - 24s 26ms/step - loss: 0.0220 - accuracy: 0.9929 Epoch 7/12
        Epoch 9/12
        Epoch 10/12
938/938 [======
                    Epoch 11/12
        938/938 [====
Epoch 12/12
                    Out[9]: <tensorflow.python.keras.callbacks.History at 0x7fb5701ef610>
In [ ]: ▶
```