Distributed Strategy

LATEST SUBMISSION GRADE

100%

1.

Question 1

Which of the following is *true* about training your model using data parallelism technique? Check all that are true.

1 / 1 point

☐

The full data set is split up and subsets of the data are stored across multiple machines

Correct

Correct! Data parallelism is meant to improve efficiency by not having to store or process all of the data on the same machine.

☐

All of the data is on 1 master machine, and copies of the data are then distributed to machines having different model architectures based on their capacity of processing the data.

☐

The same model architectures are used on different machines, and each machine processes the entire data set.

☐

Weights from different machines are aggregated and updated into a single model.

Correct

Correct! All the learnings from training on multiple machines should be used to update a single model.

2.

Question 2

In TensorFlow version 2, tf.distribute.Strategy class supports _______. Check all that apply.

1 / 1 point

☐

Eager Mode

Correct

Correct!

☐

Graph Mode

Correct

Correct!

3.

Question 3

Which of the following are *true* of both MirroredStrategy and TPU Strategy? Check all that are *true*.

1 / 1 point

☐

Variables are synchronized (mirrored) across each replica of the model

Correct

Correct! Variables are mirrored across the copies of the model.

☐

Uses a single machine

Correct

Correct! Both of these strategies use a single machine.

☐

Uses multiple machines

☐

The same model is replicated on each core.

Correct

Correct! Both of these strategies use multiple cores on the same machine (either GPU for Mirrored Strategy or TPU for TPU strategy)

4.

Question 4

To modify training code to work with Mirrored Strategy, which of the following should we do? Choose all that apply.

1 / 1 point

☐

Put code that creates the model object inside the scope of “*with strategy.scope()*”.

Correct

Correct: the model creation code should be written within the scope of the strategy.

☐

Adjust the batch size to equal the batch size per replica times the number of replicas

Correct

Correct! The batch size that the model can handle is now the number of examples that can be processed across all replicas of the model.

☐

Put the code that creates, compiles and fits the model inside the scope of “*with strategy.scope()”*.

☐

Increase the batch size as long as the number is 2^n (e.g. 64, 128, 256 etc).

5.

Question 5

To modify training code to work with distributed data, which of the following should we do? Choose all that apply.

1 / 1 point

☐

Use *strategy.reduce* to aggregate the losses across the replicas.

Correct

Correct! After the replicas all train, update their weights, and return their losses, their losses are aggregated using *strategy.reduce*

☐

Use *strategy.experimental_distribute_dataset* to convert training and test sets into distributed datasets.

Correct

Correct!

☐

Use *strategy.run* to run the code that updates the model weights (calculating loss, calculating the gradients, and applying the gradients).

Correct

Correct! Use *strategy.run* and pass in a function that contains the code which updates the model weights and returns the calculated loss.

☐

Replace the code that updates the model weights (calculating loss, calculating gradients, and applying the gradients) so that each training step handles all replicas at once.

6.

Question 6

To use the TPU strategy, there are some steps that you'll take before running the training code. Please think about which line of code implements each step and choose the set of code that performs these steps in this order.

1 Get the TPU address

2 Find the TPU cluster

3 Connect to the TPU cluster

4 Initialize the TPU cluster

5 Create your TPU strategy

1 / 1 point

1

2

3

4

5

```
tpu_address = 'grpc://' + os.environ['COLAB_TPU_ADDR']
```

```
tf.config.experimental_connect_to_cluster(tpu)
```

```
tf.distribute.cluster_resolver.TPUClusterResolver(tpu_address)
```

```
tf.tpu.experimental.initialize_tpu_system(tpu)
```

```
strategy = tf.distribute.experimental.TPUStrategy(tpu)
```

```
tpu_address = 'grpc://' + os.environ['COLAB_TPU_ADDR']
tf.distribute.cluster_resolver.TPUClusterResolver(tpu_address)
tf.config.experimental_connect_to_cluster(tpu)
tf.tpu.experimental.initialize_tpu_system(tpu)
strategy = tf.distribute.experimental.MirroredStrategy(tpu)
```

```
tpu_address = 'grpc://' + os.environ['COLAB_TPU_ADDR']
tf.distribute.cluster_resolver.TPUClusterResolver(tpu_address)
```

`tf.config.experimental_connect_to_cluster(tpu)`

`tf.tpu.experimental.initialize_tpu_system(tpu)`

`strategy = tf.distribute.experimental.TPUStrategy(tpu)`

5

`tf.tpu.experimental.initialize_tpu_system(tpu)`

Correct

Correct!