

SCA DISTRIBUTED TRAINING AI WORKSHOP DOC

**Intel oneAPI Hands-on Distributed Training workshop using
Intel oneAPI AI Analytics toolkit and Horovod**

1st March 2021

Jupyter Notebook Basics(to read)

- a) This command at the top of a particular cell will create a bash script containing all the commands of that cell in the “filename.sh” file.

```
%%writefile filename.sh
```

- b) The Job(bash script file) is submitted to run on a CPU node using the following command

```
qsub filename.sh -l nodes=1:ppn=2 -d .
```

- c) To check the status of all the jobs `qstat` is used. It shows the running jobs.
- d) After the job is finished running, it creates one output and one error file with name `filename.sh.o.` & `filename.sh.e.` with some number as suffix.

Clone github repository using following command:

```
git clone https://github.com/vdevaram/oneapi_workshop.git
```

Exercise 0

Environment set up for Distributed Training Benchmarking

- a) Open the “`ex0_environment_setup.ipynb`” and activate Python 3.7(oneAPI) kernel.
- b) Create a virtual environment “`inteltfhorovod`” with intel tensorflow 1.15.2 and horovod 0.16.1 installed in it.
- c) Intel MPI is already installed with oneAPI on DevCloud.

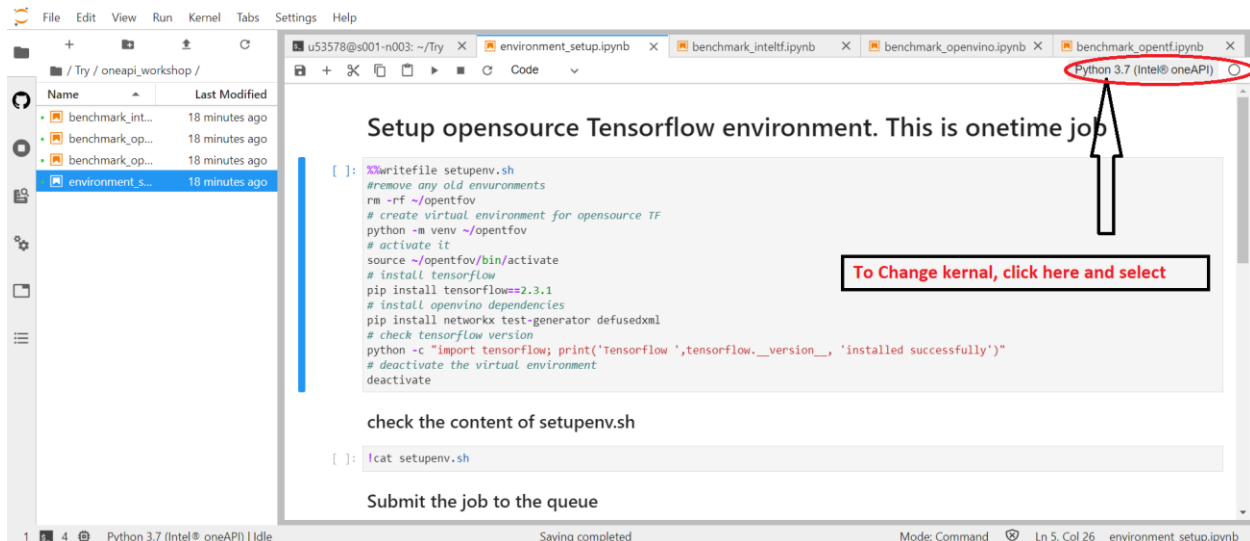


Figure: How To change the kernel

Exercise 1

Running the Resnet-50 Training with single node

- Open the “[ex1_benchmark_1node.ipynb](#)” and activate Python 3.7(oneAPI) kernel.
- Download Tensorflow Benchmarks from Github Repository.
- For non-interactive job submission, PBS commands are being used. A bash script is created for each training.
- “[#PBS -l nodes=1:ppn=2](#)” requests scheduler to assign one node with 2 processes per node.
- Submit the job “[benchmark_1node.sh](#)” containing the commands to run training benchmarks.
- After the job finishes running, note down the training and throughput values(images/sec).

Exercise 2

Running the Resnet-50 Training with two nodes

- a) Open the “[ex2_benchmark_2node.ipynb](#)” and activate Python 3.7(oneAPI) kernel.
- b) Download Tensorflow Benchmarks from Github Repository.
- c) For non-interactive job submission, PBS commands are being used. A bash script is created for each training.
- d) “[#PBS -l nodes=2:ppn=2](#)” requests scheduler to assign one node with 2 processes per node.
- e) Submit the job “benchmark_2node.sh” containing the commands to run training benchmarks.
- f) After the job finishes running, note down the training and throughput values(images/sec).

Exercise 3

Running the Resnet-50 Training with four nodes

- a) Open the “[ex1_benchmark_4node.ipynb](#)” and activate Python 3.7(oneAPI) kernel.
- b) Download Tensorflow Benchmarks from Github Repository.
- c) For non-interactive job submission, PBS commands are being used. A bash script is created for each training.
- d) “[#PBS -l nodes=4:ppn=2](#)” requests scheduler to assign 4 nodes with 2 processes per node.
- e) Submit the job “benchmark_4node.sh” containing the commands to run training benchmarks.
- f) After the job finishes running, note down the training and throughput values(images/sec).

Documentation Links:

Intel DevCloud for oneAPI – Documentation

https://devcloud.intel.com/oneapi/get_started

Intel oneAPI AI Analytics toolkit – Documentation

<https://software.intel.com/content/www/us/en/develop/tools/oneapi/ai-analytics-toolkit.html>

Intel MPI Documentation

<https://software.intel.com/content/www/us/en/develop/tools/oneapi/components/mpi-library.html>

Feedback

If you have any feedback on this document or ideas to improve it, please write an email to any of us at:

vinod.devarampati@intel.com

lakshminarasimhan.ranganathan@intel.com

aditya.sirvaiya@intel.com