How to make Jupyter Notebook to run on GPU?

Asked 4 years, 1 month ago Modified 8 months ago Viewed 169k times

63

In Google Collab you can choose your notebook to run on cpu or gpu environment. Now I have a laptop with NVDIA Cuda Compatible GPU 1050, and latest anaconda. How to have similiar feature to the collab one where I can simply make my python to run on GPU?

gpu

24

Share Improve this question Follow

asked Jun 23, 2018 at 14:18

Hari Prasad
1,441 1 13 18

6 Answers

Sorted by:
Trending sort available

Highest score (default)

45

I am answering my own question. Easiest way to do is use connect to Local Runtime (https://research.google.com/colaboratory/local-runtimes.html) then select hardware accelerator as GPU as shown in (https://medium.com/deep-learning-turkey/google-colab-free-gpu-tutorial-e113627b9f5d).

Share Improve this answer Follow

answered Jun 23, 2018 at 15:29

Hari Prasad
1,441 1 13 18

5 Not sure why it is downvoted. It is a important question and answer for anyone starting the GPU journey in to Analytics. – Hari Prasad Jun 27, 2018 at 8:32

18 Did you find a way to enable your GPU for a notebook on your local machine? – capm Jun 2, 2019 at 7:03

Once you set up the GPU with Colab, then you can run the codes on Jupyter right? – Arnab Sinha Jun 5, 2019 at 7:32

2 How could run python code in jupyter notebook using GPU? It's possible to set and choose GPU like Google Colab? – BarzanHayati Aug 4, 2019 at 12:47

2 It's a nice and easy way but which forces you to upload notebooks on GDrive and keep things in sync which can be inconvenient (especially when you have a bunch of files to use). – Maxie Berkmann Mar 3, 2020 at 11:41

1. Install Miniconda/anaconda

11

2. Download CUDA Toolkit (acc to OS)

   Follow this (for LINUX CUDA Toolkit):

```
 a. Wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin

 b. sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-pin-600

 c. wget https://developer.download.nvidia.com/compute/cuda/11.0.3/local_installers/cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb

 d. sudo dpkg -i cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb

 e. sudo apt-key add /var/cuda-repo-ubuntu2004-11-0-local/7fa2af80.pub

 f. sudo apt-get update

 g. sudo apt-get -y install cuda
```

3. Download and install cuDNN (create NVIDIA acc)

   a. Paste the cuDNN files(bin,include,lib) inside CUDA Toolkit Folder.

4. Add CUDA path to ENVIRONMENT VARIABLES (see a tutorial if you need.)

5. Create an environment in miniconda/anaconda

```
Conda create -n tf-gpu

Conda activate tf-gpu

pip install tensorflow-gpu
```

6. Install Jupyter Notebook (JN)

```
pip install jupyter notebook
```

7. DONE! Now you can use tf-gpu in JN.

Share Improve this answer Follow

answered Aug 7, 2020 at 17:03

Manmohan Dogra
129 1 4

4

I've written a medium article about how to set up Jupyterlab in Docker (and Docker Swarm) that accesses the GPU via CUDA in PyTorch or Tensorflow.

Set up your own GPU-based Jupyter

I'm clear that you don't search for a solution with Docker, however, it saves you a lot of time when using an existing Dockerfile with plenty of packages required for statistics and ML.

Share Improve this answer Follow

answered Dec 27, 2019 at 17:56


Christoph Schranz
684 6 6

How compatible would that be for Jupyter Notebook? I've never used Jupyterlab so I don't know how it would differ. – Maxie Berkmann Feb 25, 2020 at 14:31

2 I hope this link helps. Jupyterlab has a better UI, both are based on IPython, so they will have a similar performance. – Christoph Schranz Feb 25, 2020 at 20:56

0

I have OpenCL SDK for Intel setup for my Windows 10, 64 bit system. I have also installed PyOpenCL for Python 3.7. I didn't install it with conda but pip with the WHL file. I can use it with IDEL with no problem. To use PyOpenCL with Jupyter notebook and Spyder (Anaconda3). I did further with the following:

1. Find **Anaconda Powershell Prompt (Anaconda3)** from Windows start menu and **run it as administrator** (to avoid user permission error.)
2. Try and update like so:

```
(base) PS C:\WINDOWS\system32> conda update -n base conda -c anaconda
```

(Warning: this may take some time if it has not been updated for some time..) type in **y** to continue when asked.

Given that is done with no error, now you are ready to install PyOpenCL:

```
(base) PS C:\WINDOWS\system32> conda install -c conda-forge pyopencl
```

Enter **y** to proceed when asked.

(This will be quick!)

Now you can start Spyder or Jupyter to test it.

```
import pyopencl as cl
```

Giving no error, you are all set! And that is. It has been tested working with Jupyter and Spyder 3 on Windows 10, 64 bit. I hope you will find this helpful.

Share Improve this answer Follow

answered Apr 15, 2020 at 4:32


Harry
1,009 12 13

0

Before following below steps make sure that below pre-requisites are in place:

1. Python 3.x is installed.
2. Anaconda is installed.
3. CUDA toolkit is installed.

Steps to run Jupyter Notebook on GPU

1. Create a new environment using Conda:

Open a command prompt with admin privilege and run the below command to create a new environment with the name gpu2.

```
Conda create -n gpu2 python=3.6
```

Follow the on-screen instructions as shown below and gpu2 environment will be created. enter image description here enter image description here

- Run below command to list all available environments.

```
conda info -e
```

- Now, run below command to activate / enable newly created gpu2 environment.

```
conda activate -n gpu2
```

Install tensorflow-gpu. Here I have installed tensorflow-gpu v2.3.0. You can check below link to find the compatibale tensorflow-gpu version with your install Python version.

https://www.tensorflow.org/install/source_windows#tested_build_configurations

```
pip install tensorflow-gpu==2.3.0
```

follow the instructions here[https://www.techentice.com/how-to-make-jupyter-notebook-to-run-on-gpu/]

Share Improve this answer Follow

answered Aug 5, 2021 at 9:32

Kamyab
5 4

-2

Stop wasting your time and follow these steps:

1-Go to `https://www.tensorflow.org/install/source#gpu`

2-Copy to a textfile the following, the latest TensorFlow version, Python version, cuDNN and CUDA. At the current date of writing this comment (3rd of Dec/2021), the latest releases are:

TensorFlow version= `tensorflow-2.7.0`

Python version = `3.7-3.9` -> here `7-3` means releases `3 or 4 or 5 or 6 or 7`

cuDNN= `8.1`

CUDA= `11.2`

3- I assume that you have already installed anaconda, if not ask uncle google.

4- Open anaconda prompt and run the following commands:

```
conda create --name my_env python=3.7.9
```

This will create a new python environment other than your root/base environment. Remember `my_env` can be changed to any names. As for `python=3.7.9`, `cuDNN=8.1` and `CUDA=11.2` versions(numbers) must be changed to whatever release you find on the tensorflow website.

List the current environments you have:

```
conda env list
```

Note the `*` beside your `base` and `(base)` means that you are running your base environment. Thus, we need to change to the newly created environment by typing

```
activate my_env
```

Now, we need to install cudatoolkit:

```
conda install cudatoolkit=11.2
```

This will install the latest Cuda version of `11.2` which is probably `11.2.2` so don't freak out.

We will also need to install cuDNN:

```
conda install cudnn=8.1
```

Then, install the required version of tensorflow

```
conda install tensorflow-gpu==2.7.0
```

Now type `jupyter` to launch jupyter notebook in your newly created `my_env`. Then type `import tensorflow as tf` and run in the first cell then `tf.test.is_gpu_available()` and run in the second cell. If the output is `true` then you are good to go otherwise something went wrong. Of course, there are lots of checks and methods to perform but it seems this is the fastest and simplest.

Don't forget to subscribe and share

Thanks

Share Improve this answer Follow

answered Dec 3, 2021 at 5:39

E.Zolduoarrati
1,271 1 8 8

1 You said " `3.7-3.9` here `7-3` means `3 or 4 or 5 or 6 or 7` ". It certainly does not. `3.7-3.9` means either 3.7 or 3.8 or 3.9. – Tim Roberts Dec 9, 2021 at 20:12