Docker

Docker (https://docs.docker.com/install/) uses *containers* to create virtual environments that isolate a TensorFlow installation from the rest of the system. TensorFlow programs are run *within* this virtual environment that can share resources with its host machine (access directories, use the GPU, connect to the Internet, etc.). The TensorFlow Docker images (https://hub.docker.com/r/tensorflow/tensorflow/) are tested for each release.

Docker is the easiest way to enable TensorFlow GPU support (https://www.tensorflow.org/install/gpu) on Linux since only the NVIDIA® GPU driver (https://github.com/NVIDIA/nvidia-docker/wiki/Frequently-Asked-Questions#how-do-i-install-the-nvidia-driver) is required on the *host* machine (the *NVIDIA® CUDA® Toolkit* does not need to be installed).

TensorFlow Docker requirements

1. Install Docker (https://docs.docker.com/install/) on your local *host* machine.
2. For GPU support on Linux, install NVIDIA Docker support (https://github.com/NVIDIA/nvidia-docker).
    - Take note of your Docker version with `docker -v`. Versions **earlier than** 19.03 require nvidia-docker2 and the `--runtime=nvidia` flag. On versions **including and after** 19.03, you will use the `nvidia-container-toolkit` package and the `--gpus all` flag. Both options are documented on the page linked above.

To run the **`docker`** command without **`sudo`**, create the **`docker`** group and add your user. For details, see the post-installation steps (https://docs.docker.com/install/linux/linux-postinstall/).

Download a TensorFlow Docker image

The official TensorFlow Docker images are located in the tensorflow/tensorflow (https://hub.docker.com/r/tensorflow/tensorflow/) Docker Hub repository. Image releases are tagged (https://hub.docker.com/r/tensorflow/tensorflow/tags/) using the following format:

Tag	Description
`latest`	The latest release of TensorFlow CPU binary image. Default.
`nightly`	Nightly builds of the TensorFlow image. (Unstable.)
`version`	Specify the *version* of the TensorFlow binary image, for example: *2.1.0*
`devel`	Nightly builds of a TensorFlow `master` development environment. Includes TensorFlow source code.

Tag	Description
`custom-op`	Special experimental image for developing TF custom ops. More info here (https://github.com/tensorflow/custom-op).

Each base *tag* has variants that add or change functionality:

Tag Variants	Description
`tag-gpu`	The specified *tag* release with GPU support. (See below (#gpu_support))
`tag-jupyter`	The specified *tag* release with Jupyter (includes TensorFlow tutorial notebooks)

You can use multiple variants at once. For example, the following downloads TensorFlow release images to your machine:

```
$ docker pull tensorflow/tensorflow                     # latest stable release
$ docker pull tensorflow/tensorflow:devel-gpu           # nightly dev release w/ GPU support
$ docker pull tensorflow/tensorflow:latest-gpu-jupyter  # latest release w/ GPU support and Jup
```

Start a TensorFlow Docker container

To start a TensorFlow-configured container, use the following command form:

```
$ docker run [-it] [--rm] [-p hostPort:containerPort] tensorflow/tensorflow[:tag] [command]
```

For details, see the docker run reference (https://docs.docker.com/engine/reference/run/).

Examples using CPU-only images

Let's verify the TensorFlow installation using the `latest` tagged image. Docker downloads a new TensorFlow image the first time it is run:

```
$ docker run -it --rm tensorflow/tensorflow \
   python -c "import tensorflow as tf; print(tf.reduce_sum(tf.random.normal([1000, 1000])))"
```

ss: TensorFlow is now installed. Read the tutorials (https://www.tensorflow.org/tutorials) to get started.

Let's demonstrate some more TensorFlow Docker recipes. Start a `bash` shell session within a TensorFlow-configured container:

```
$ docker run -it tensorflow/tensorflow bash
```

Within the container, you can start a **`python`** session and import TensorFlow.

To run a TensorFlow program developed on the *host* machine within a container, mount the host directory and change the container's working directory (**`-v hostDir:containerDir -w workDir`**):

```
$ docker run -it --rm -v $PWD:/tmp -w /tmp tensorflow/tensorflow python ./script.py
```

Permission issues can arise when files created within a container are exposed to the host. It's usually best to edit files on the host system.

Start a Jupyter Notebook (https://jupyter.org/) server using TensorFlow's nightly build:

```
$ docker run -it -p 8888:8888 tensorflow/tensorflow:nightly-jupyter
```

Follow the instructions and open the URL in your host web browser: **`http://127.0.0.1:8888/?token=...`**

GPU support

Docker is the easiest way to run TensorFlow on a GPU since the *host* machine only requires the NVIDIA® driver (https://github.com/NVIDIA/nvidia-docker/wiki/Frequently-Asked-Questions#how-do-i-install-the-nvidia-driver) (the *NVIDIA® CUDA® Toolkit* is not required).

Install the Nvidia Container Toolkit (https://github.com/NVIDIA/nvidia-docker/blob/master/README.md#quickstart) to add NVIDIA® GPU support to Docker. **`nvidia-container-runtime`** is only available for Linux. See the **`nvidia-container-runtime`** platform support FAQ (https://github.com/NVIDIA/nvidia-docker/wiki/Frequently-Asked-Questions#platform-support) for details.

Check if a GPU is available:

```
$ lspci | grep -i nvidia
```

Verify your **`nvidia-docker`** installation:

```
$ docker run --gpus all --rm nvidia/cuda nvidia-smi
```

`nvidia-docker` v2 uses **`--runtime=nvidia`** instead of **`--gpus all`**. **`nvidia-docker`** v1 uses the **`nvidia-docker`** alias, rather tha
`time=nvidia` or **`--gpus all`** command line flags.

Examples using GPU-enabled images

Download and run a GPU-enabled TensorFlow image (may take a few minutes):

```
$ docker run --gpus all -it --rm tensorflow/tensorflow:latest-gpu \
   python -c "import tensorflow as tf; print(tf.reduce_sum(tf.random.normal([1000, 1000])))"
```

It can take a while to set up the GPU-enabled image. If repeatedly running GPU-based scripts, you can use `docker exec` to reuse a container.

Use the latest TensorFlow GPU image to start a `bash` shell session in the container:

```
$ docker run --gpus all -it tensorflow/tensorflow:latest-gpu bash
```

ss: TensorFlow is now installed. Read the tutorials (https://www.tensorflow.org/tutorials) to get started.

Last updated 2021-01-28 UTC.