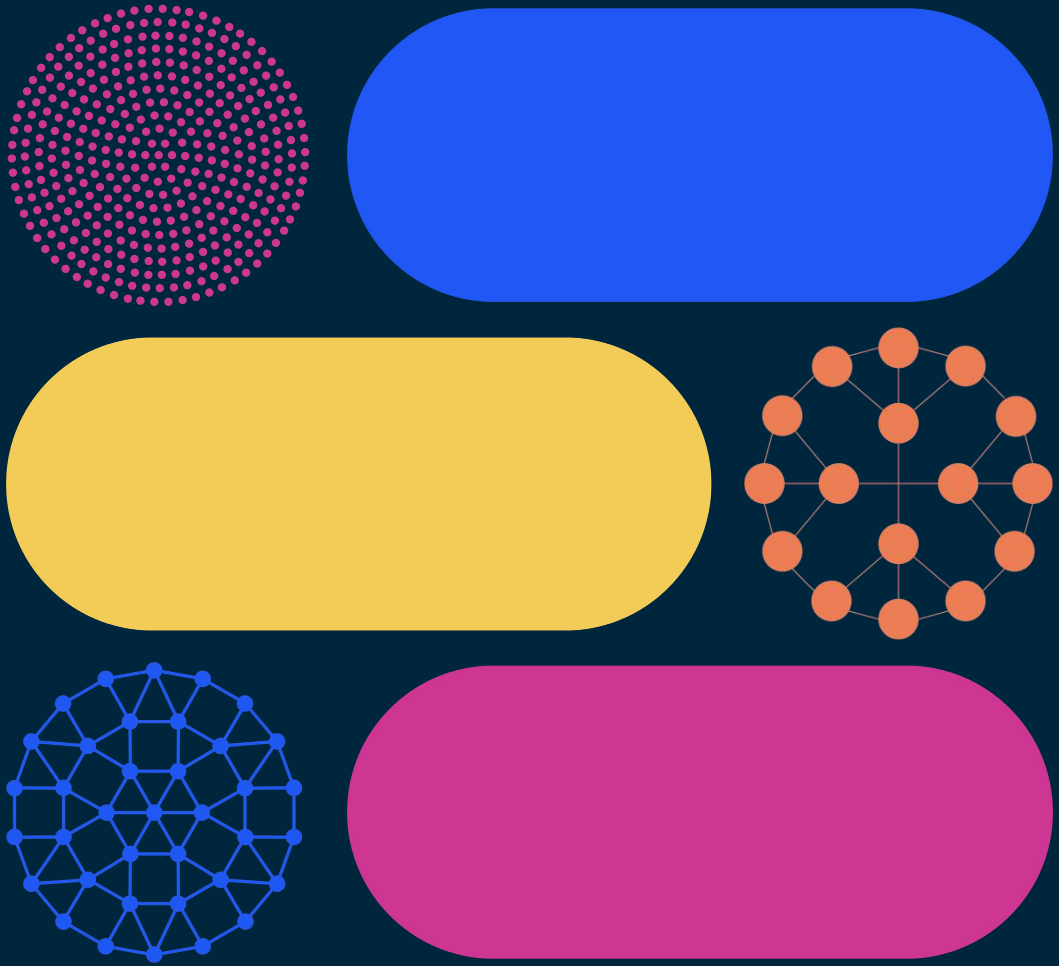


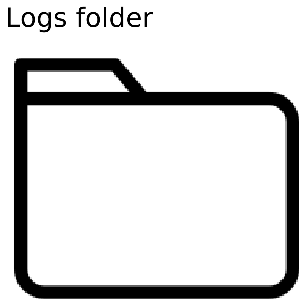
Tensorboard with run:ai

(using ResNet examples)

Tensorboard overview

What is needed for Tensorboard to run?

1) A **logs folder** to store objects related to the runs



How does Tensorboard work?

1) To **view the Tensorboard UI**, we start the server with a CLI command

You must specify the location of the logs folder with the logdir argument. You can also set the host IP and port.

```
tensorboard \
    --logdir=/abs/path/to/logs_folder \
    --host=0.0.0.0 \
    --port=6006
```

How does Tensorboard work?

2) In order to **write records to the Tensorboard folder**, create a Tensorboard callback in our script, and pass it to a model

Note:
It is not necessary to start the Tensorboard server in order to write records.
Starting the server is only needed to view the UI.

```
import tensorflow as tf

tb_dir = "/abs/path/to/logs_folder"

“””
code to build and compile your model
“””

tensorboard_callback = tf.keras.callbacks.TensorBoard(
    log_dir=tb_dir)

history = model.fit(train_ds,
                    epochs=5,
                    callbacks=[tensorboard_callback])
```

Tensorboard with run:ai

What is needed for Tensorboard to run on run:ai?

1) A **persistent** directory to keep
- Tensorboard logs folder

2) A **docker image** with the following installed
- Tensorboard
- jupyterlab*
- jupyter-server-proxy*

*needed to access the Tensorboard UI

3) A **docker image** with the following installed
- Tensorflow**
- Keras**

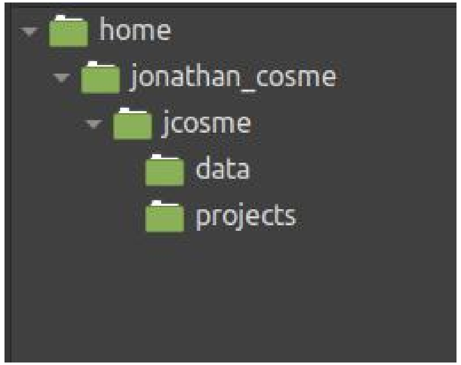
**needed in order train Tensorflow models (ResNet in our example)

Creating persistent directory

Before we start

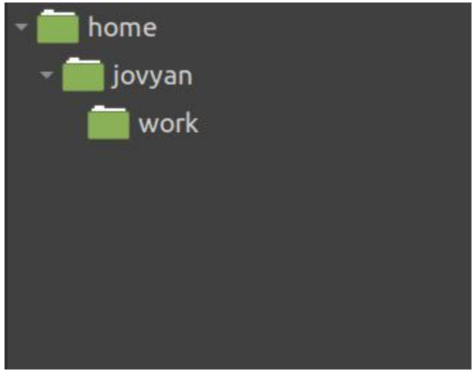
We need to create a 'tensorboard_logs' folder on our NFS.

This is what our NFS folder structure looks like now



```
home
  jonathan_cosme
    jcosme
      data
      projects
```

This is the default folder structure for our jupyter lab image

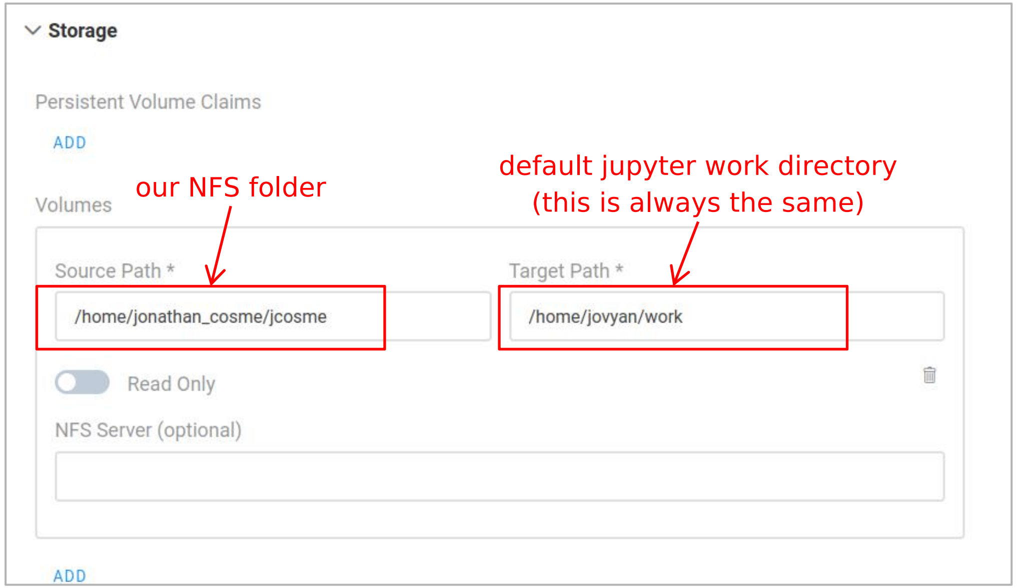


```
home
  jovyan
    work
```

Before we start

We need to create an ‘tensorboard_logs’ folder on our NFS.

For *our example:*
Whenever we create a job on run:ai, we *must always* mount our NFS to the default jupyter work directory



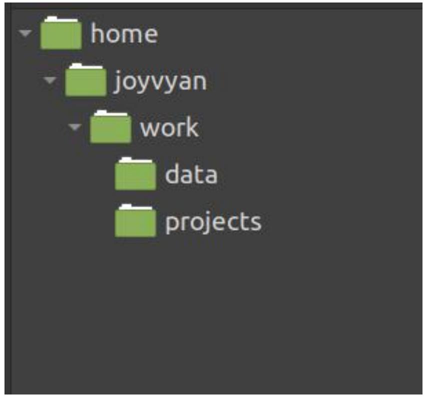
Source Path *: /home/jonathan_cosme/jcosme

Target Path *: /home/jovyan/work

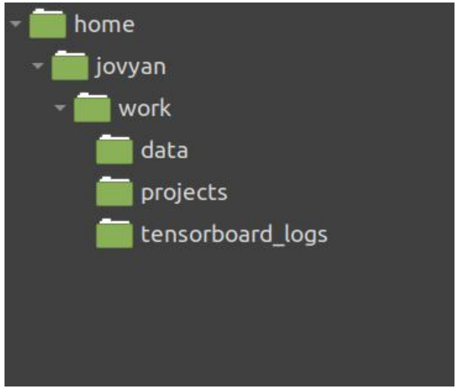
Before we start

We need to create an ‘tensorboard_logs’ folder on our NFS.

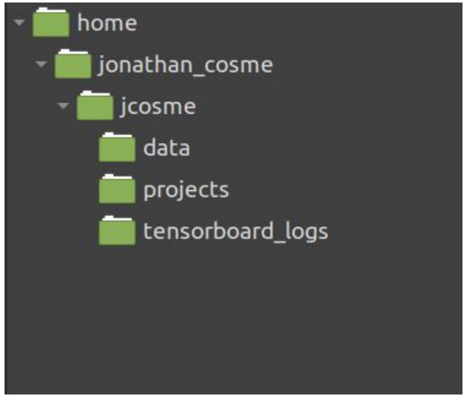
1. After we mount our NFS volume,
our work directory will look like this:



2. Using Jupyter Lab, we create an ‘tensorboard_logs’ folder within the work directory:



3. This will cause our NFS directory to automatically look like this:

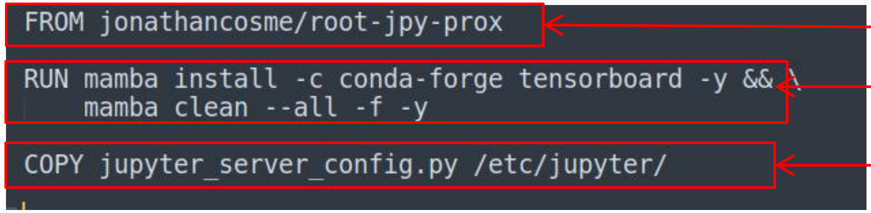


Docker images

First docker image used in our example

The docker image we will use to access the UI is:
jonathancosme/tensorboard-ui

This is what is in the dockerfile:



```
FROM jonathancosme/root-jpy-prox

RUN mamba install -c conda-forge tensorboard -y && \
    mamba clean --all -f -y

COPY jupyter_server_config.py /etc/jupyter/
```

notebook with jupyter-server-proxy already installed

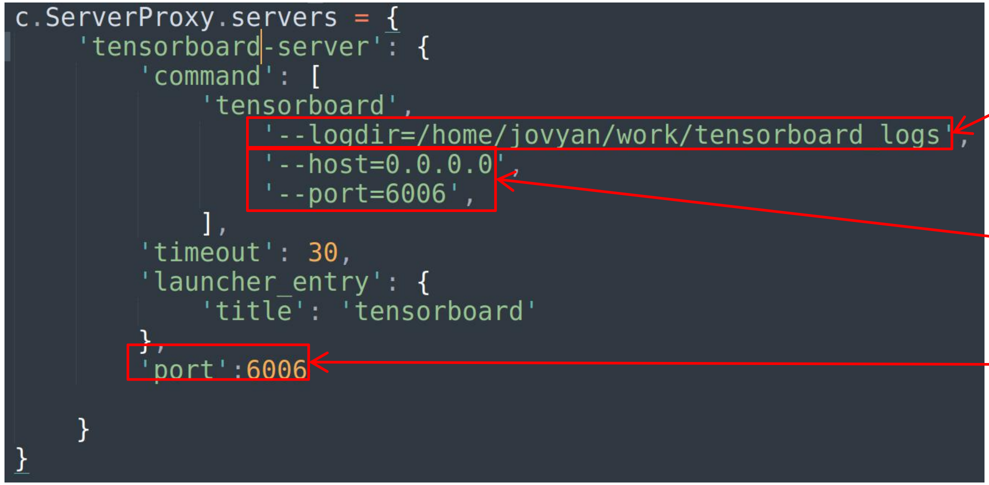
install Tensorboard

copy new config file, with Tensorboard UI access configured

First docker image used in our example

The docker image we will use to access the UI is:
jonathancosme/tensorboard-ui

In order to access the Tensorboard UI, we need to **add this entry to the jupyter_server_config.py** file, and replace the existing file in the image



```
c.ServerProxy.servers = {
    'tensorboard-server': {
        'command': [
            'tensorboard',
                '--logdir=/home/jovyan/work/tensorboard_logs',
                '--host=0.0.0.0',
                '--port=6006',
            ],
        'timeout': 30,
        'launcher_entry': {
            'title': 'tensorboard'
        },
        'port':6006

    }
}
```

we specify our Tensorboard logs folder locations (this is why we must always mount our NFS directory to the default jupyter work directory)

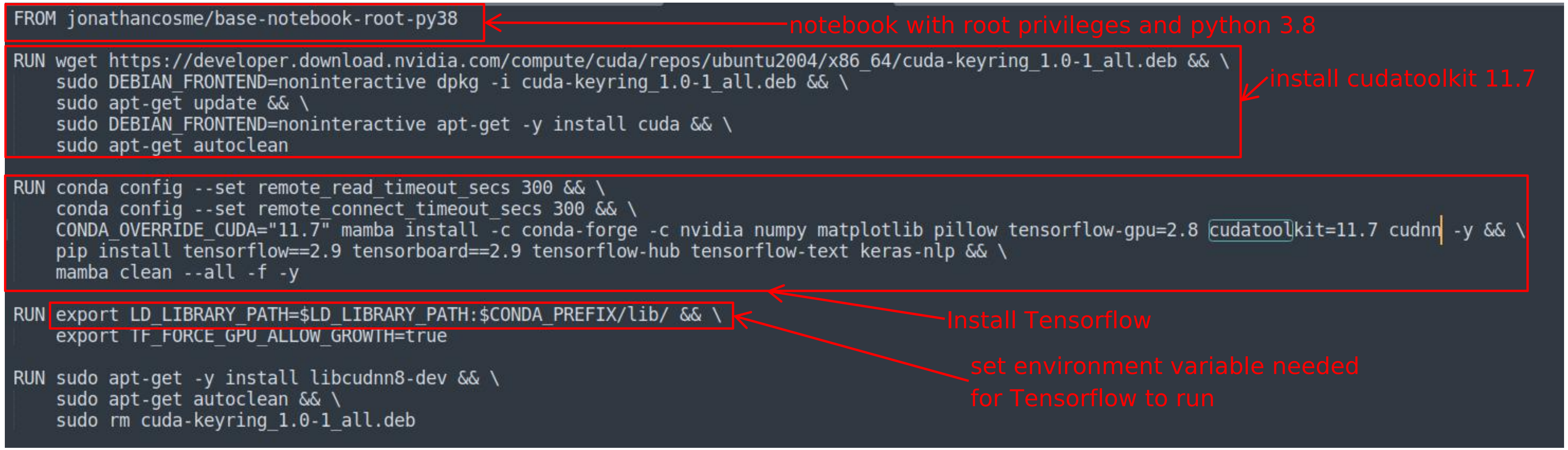
we must make sure to start the server on this IP and port

This tells jupyter to forward port 6006 to the URL

Second docker image used in our example

The docker image we will use to train a Tensorflow ResNet model is:
jonathancosme/keras-nb

This is what is in the dockerfile:



notebook with root privileges and python 3.8
install cudatoolkit 11.7
Install Tensorflow
set environment variable needed for Tensorflow to run

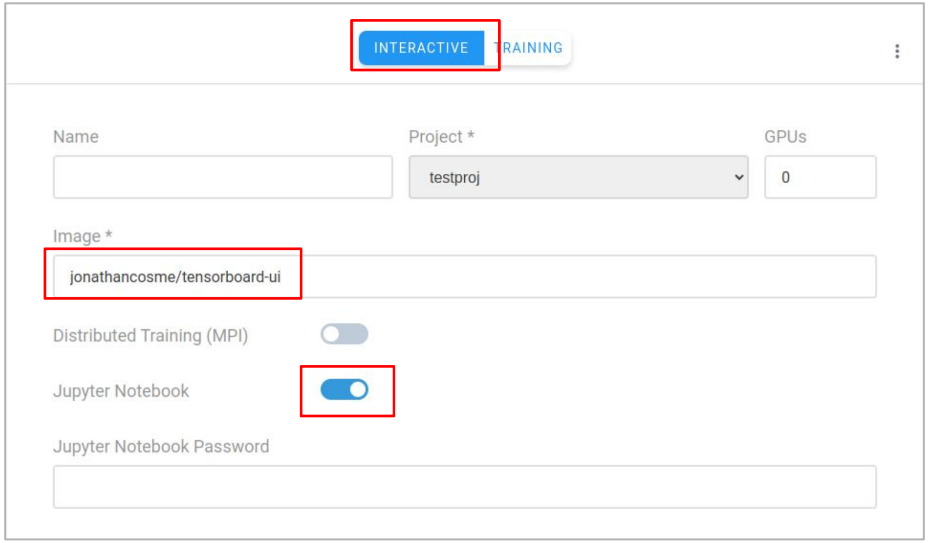
```
FROM jonathancosme/base-notebook-root-py38

RUN wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/cuda-keyring_1.0-1_all.deb && \
    sudo DEBIAN_FRONTEND=noninteractive dpkg -i cuda-keyring_1.0-1_all.deb && \
    sudo apt-get update && \
    sudo DEBIAN_FRONTEND=noninteractive apt-get -y install cuda && \
    sudo apt-get autoclean

RUN conda config --set remote_read_timeout_secs 300 && \
    conda config --set remote_connect_timeout_secs 300 && \
    CONDA_OVERRIDE_CUDA="11.7" mamba install -c conda-forge -c nvidia numpy matplotlib pillow tensorflow-gpu=2.8 cudatoolkit=11.7 cudnn -y && \
    pip install tensorflow==2.9 tensorboard==2.9 tensorflow-hub tensorflow-text keras-nlp && \
    mamba clean --all -f -y

RUN export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CONDA_PREFIX/lib/ && \
    export TF_FORCE_GPU_ALLOW_GROWTH=true

RUN sudo apt-get -y install libcudnn8-dev && \
    sudo apt-get autoclean && \
    sudo rm cuda-keyring_1.0-1_all.deb
```

Accessing the Tensorboard UI

Access Tensorboard UI

Create a jupyter interactive job with:

- image jonathancosme/tensorboard-iu
- mounted NFS folder (with ‘tensorboard_logs’ folder) in default jupyter work directory



INTERACTIVE TRAINING

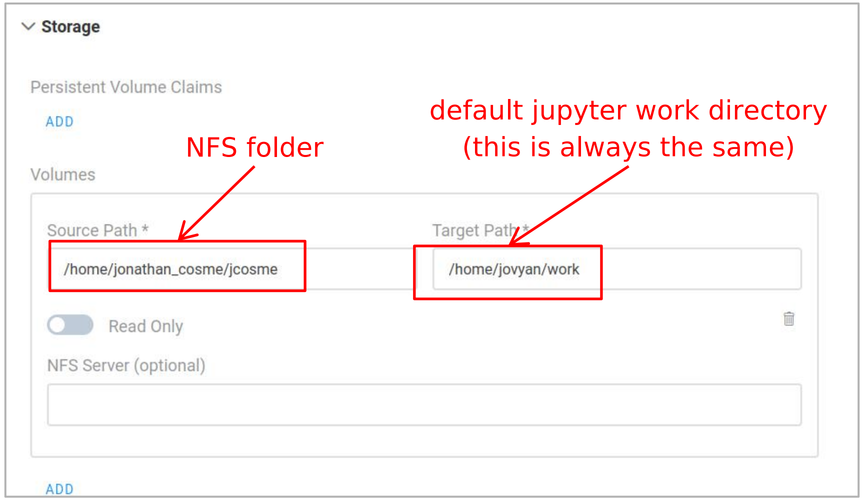
Name	Project *	GPUs
	testproj	0

Image *: jonathancosme/tensorboard-ui

Distributed Training (MPI): off

Jupyter Notebook: on

Jupyter Notebook Password:



Storage

Persistent Volume Claims

ADD

Volumes

Source Path *: /home/jonathan_cosme/jcosme

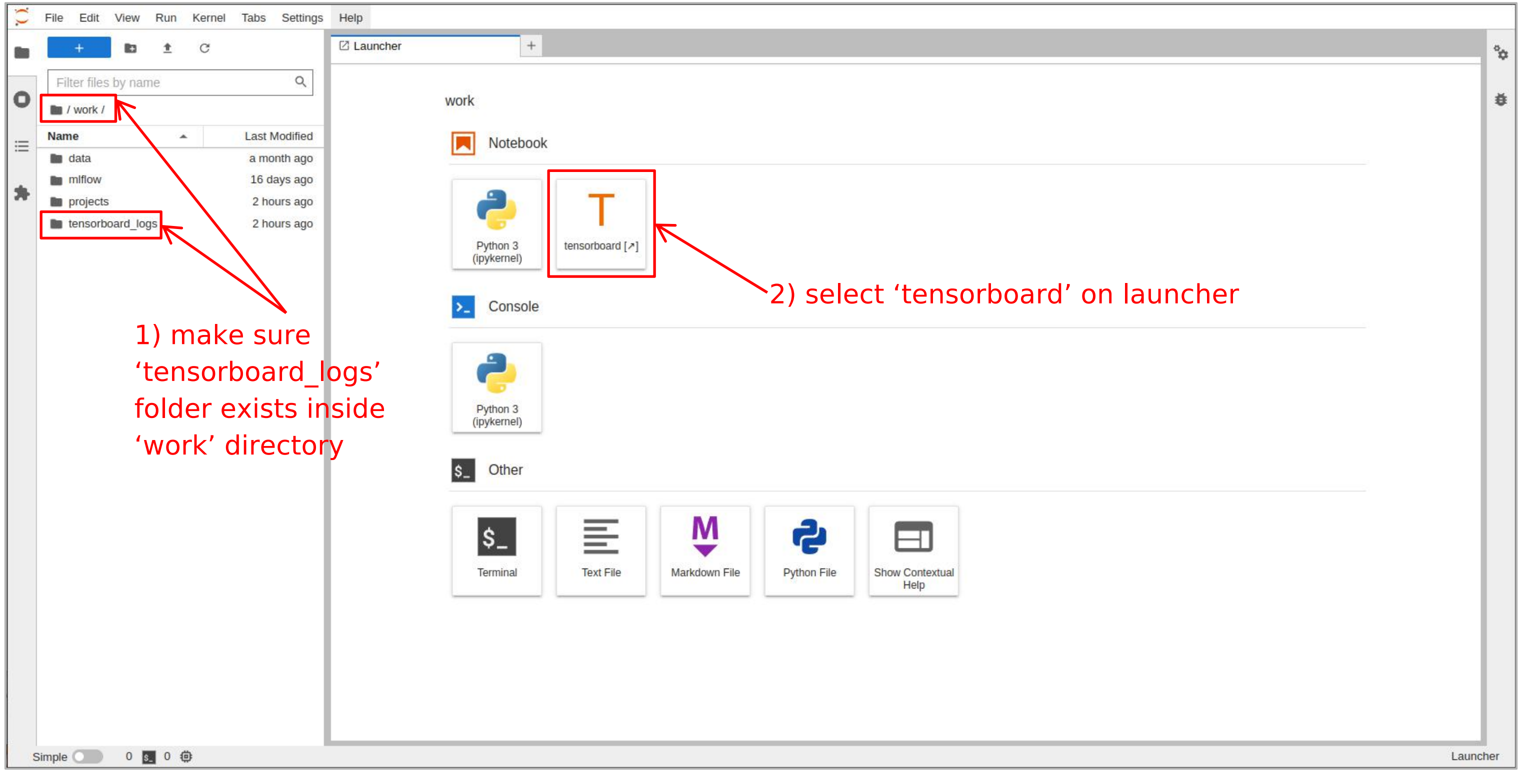
Target Path *: /home/jovyan/work

Read Only: off

NFS Server (optional):

ADD

Access Tensorboard UI



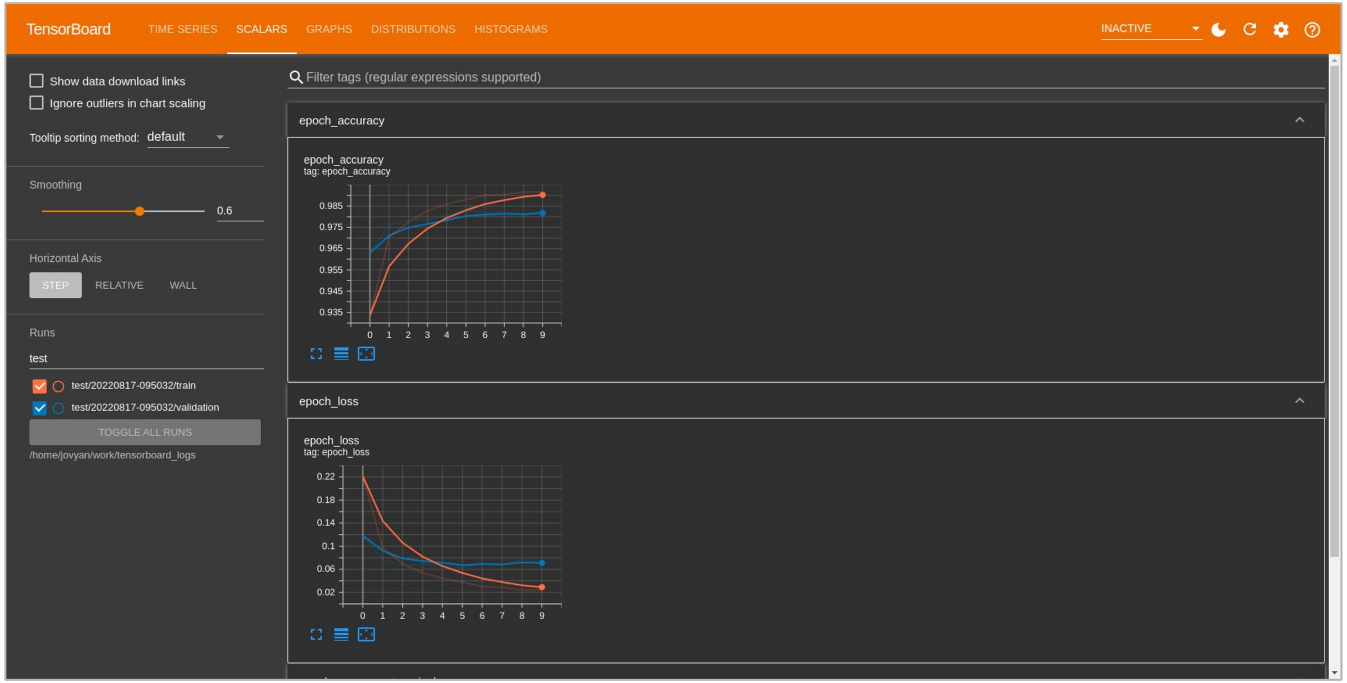
1) make sure 'tensorboard_logs' folder exists inside 'work' directory

2) select 'tensorboard' on launcher

github.com/jonathancosme/runai_tensorboard_resnet_demo

Access Tensorflow UI

A new tab should appear with the Tensorflow UI



Show data download links
Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing 0.6
Horizontal Axis: STEP RELATIVE WALL
Runs
test
test/20220817-095032/train
test/20220817-095032/validation
/home/jovyan/work/tensorboard_logs

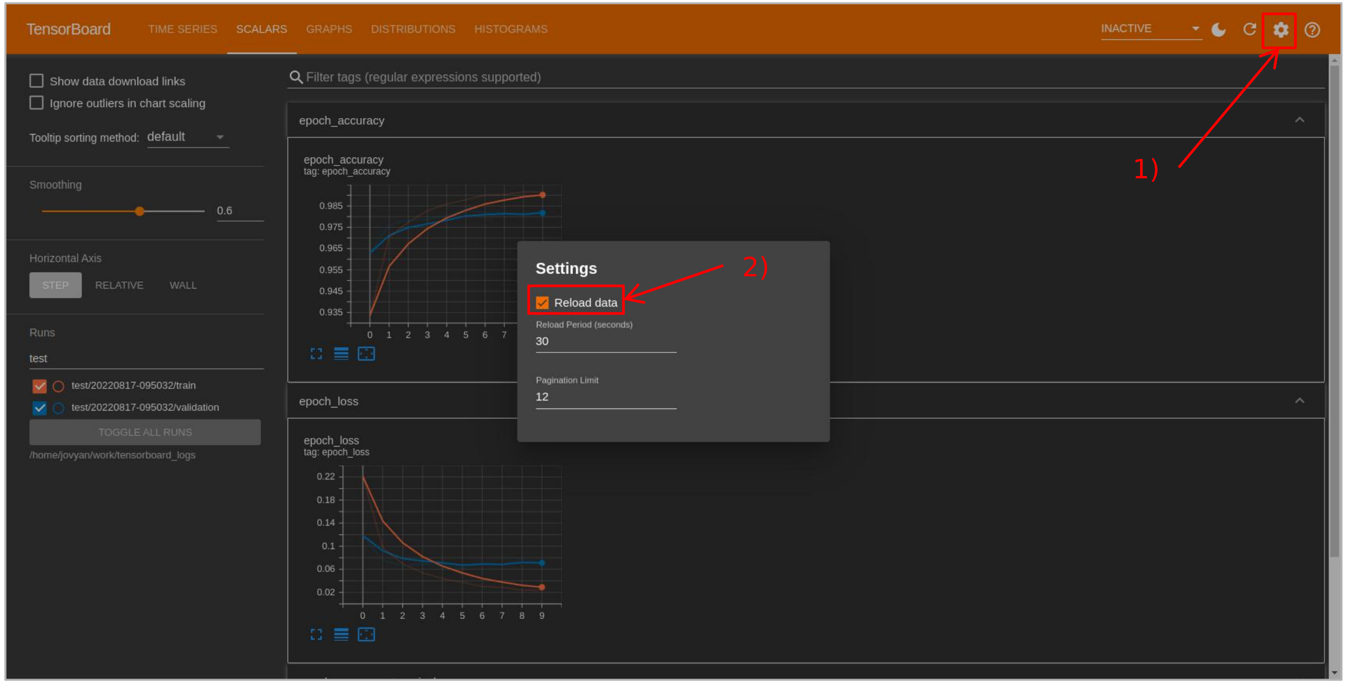
epoch_accuracy
tag: epoch_accuracy

epoch_loss
tag: epoch_loss

github.com/jonathancosme/runai_tensorboard_resnet_demo

Access Tensorflow UI

Select 'Reload data' under settings.



Show data download links

Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing 0.6

Horizontal Axis: STEP RELATIVE WALL

Runs: test

✓ test/20220817-095032/train

✓ test/20220817-095032/validation

/home/jovyan/work/tensorboard_logs

Filter tags (regular expressions supported)

Settings

✓ Reload data

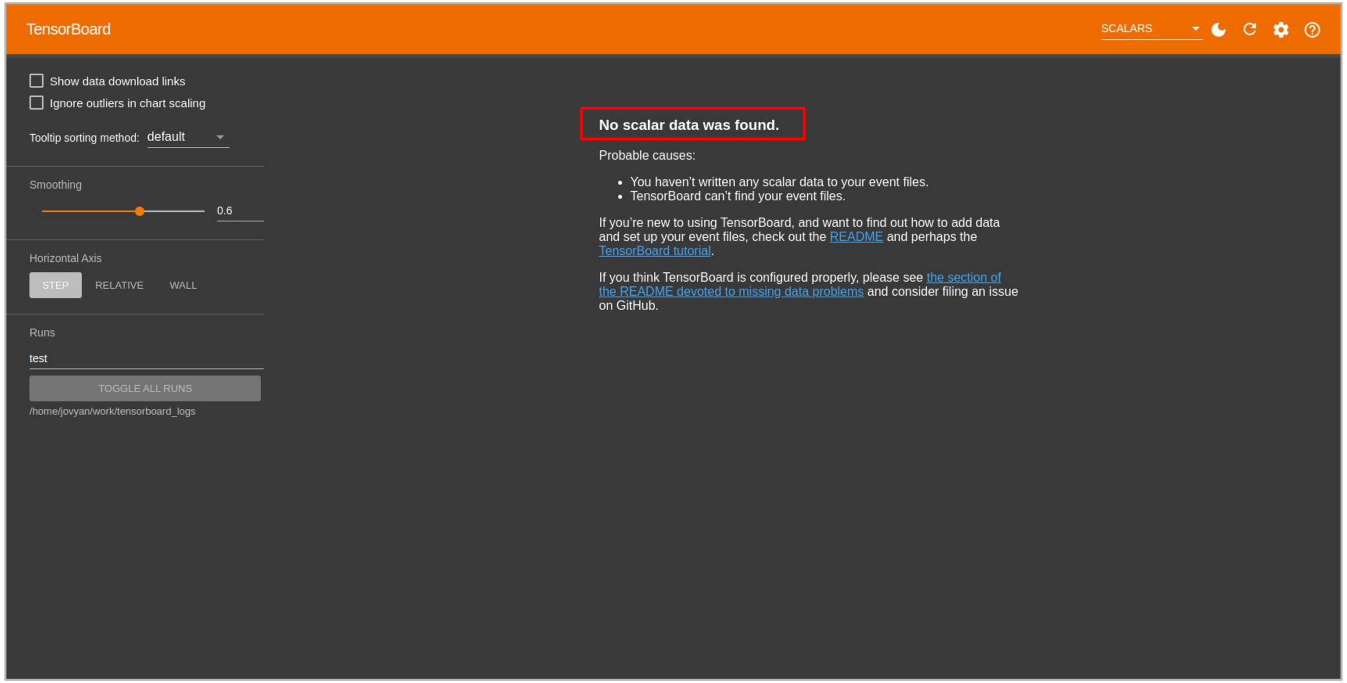
Reload Period (seconds): 30

Pagination Limit: 12

github.com/jonathancosme/runai_tensorboard_resnet_demo

Access Tensorflow UI

Note:
The first time you access the UI, there will be no data available



Show data download links
Ignore outliers in chart scaling
Tooltip sorting method: default

Smoothing
0.6

Horizontal Axis
STEP RELATIVE WALL

Runs
test
TOGGLE ALL RUNS
/home/jovyan/work/tensorboard_logs

No scalar data was found.

Probable causes:

- You haven't written any scalar data to your event files.
- TensorBoard can't find your event files.

If you're new to using TensorBoard, and want to find out how to add data and set up your event files, check out the README and perhaps the TensorBoard tutorial.

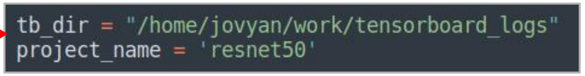
If you think TensorBoard is configured properly, please see the section of the README devoted to missing data problems and consider filing an issue on GitHub.

github.com/jonathancosme/runai_tensorboard_resnet_demo

running Tensorflow ResNet experiments with run:ai

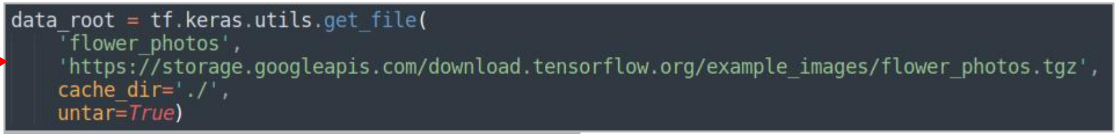
Python scripts (ResNet50 example)

Set the absolute path of the logs folder, and set a project name.



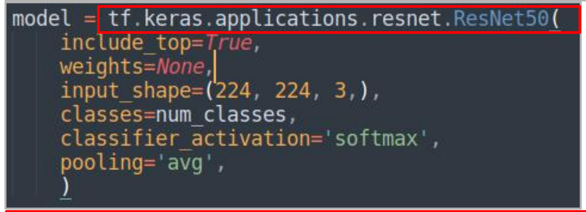
```
tb_dir = "/home/jovyan/work/tensorboard_logs"
project_name = 'resnet50'
```

Load the flowers dataset from Tensorflow



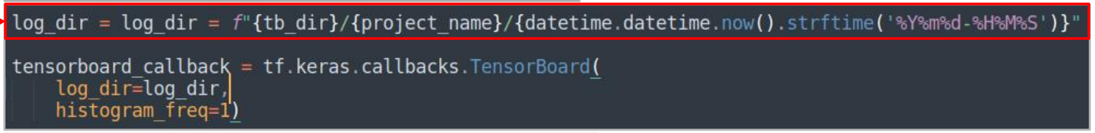
```
data_root = tf.keras.utils.get_file(
    'flower_photos',
    'https://storage.googleapis.com/download.tensorflow.org/example_images/flower_photos.tgz',
    cache_dir='./',
    untar=True)
```

Load the ResNet architecture directly from Tensorflow



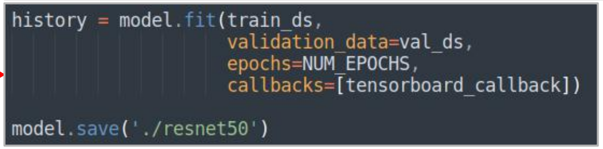
```
model = tf.keras.applications.resnet.ResNet50(
    include_top=True,
    weights=None,
    input_shape=(224, 224, 3,),
    classes=num_classes,
    classifier_activation='softmax',
    pooling='avg',
    )
```

Create a subdirectory in our tensorboard_logs folder, using project_name, then create another subfolder using the date and time



```
log_dir = log_dir = f"{tb_dir}/{project_name}/{datetime.datetime.now().strftime('%Y%m%d-%H%M%S')}"

tensorboard_callback = tf.keras.callbacks.TensorBoard(
    log_dir=log_dir,
    histogram_freq=1)
```

Pass in the callback, fit and save the model

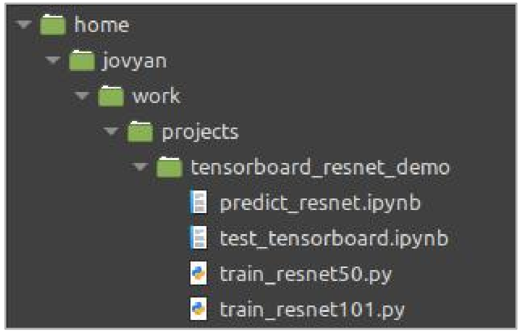


```
history = model.fit(train_ds,
                    validation_data=val_ds,
                    epochs=NUM_EPOCHS,
                    callbacks=[tensorboard_callback])

model.save('./resnet50')
```

github.com/jonathancosme/runai_tensorboard_resnet_demo

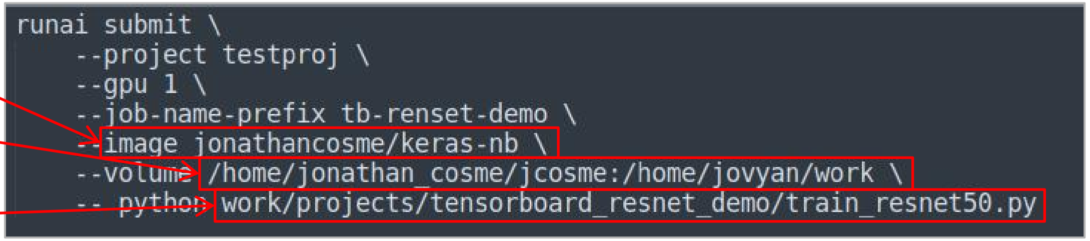
CLI submission

Our example scrips are located here:



```
home
  jovyan
    work
      projects
        tensorboard_resnet_demo
          predict_resnet.ipynb
          test_tensorboard.ipynb
          train_resnet50.py
          train_resnet101.py
```

so our CLI command would look like this:



```
runai submit \
    --project testproj \
    --gpu 1 \
    --job-name-prefix tb-renset-demo \
    --image jonathancosme/keras-nb \
    --volume /home/jonathan_cosme/jcosme:/home/jovyan/work \
    -- python work/projects/tensorboard_resnet_demo/train_resnet50.py
```

make sure you use the keras-nb docker image

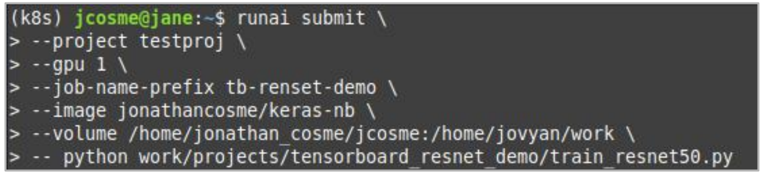
make sure you mount the NFS to the work directory

The code to run the job must specify the python script with relative location

github.com/jonathancosme/runai_tensorboard_resnet_demo

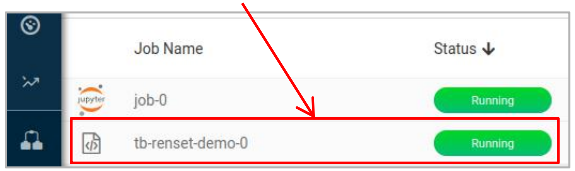
Example job submission

1) submit CLI command



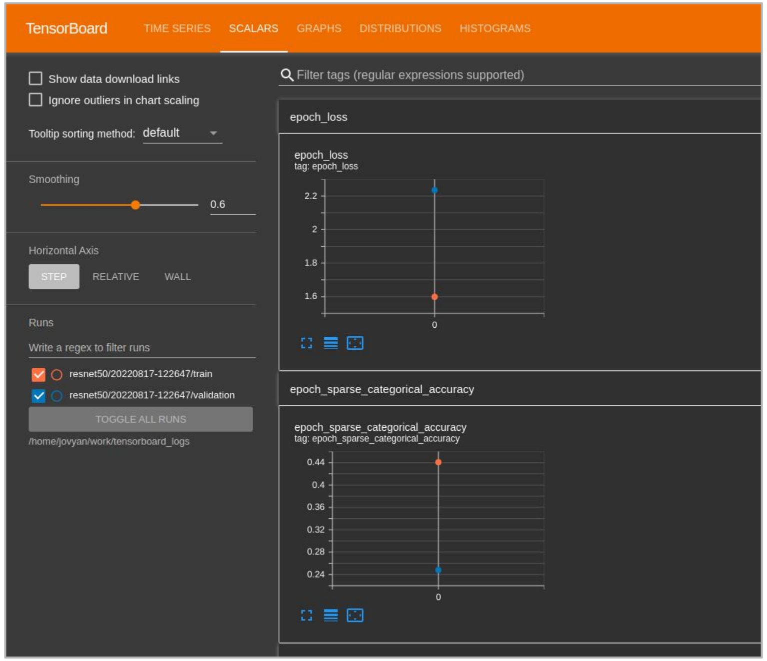
```
(k8s) jcosme@jane:~$ runai submit \
> --project testproj \
> --gpu 1 \
> --job-name-prefix tb-renset-demo \
> --image jonathancosme/keras-nb \
> --volume /home/jonathan_cosme/jcosme:/home/jovyan/work \
> -- python work/projects/tensorboard_resnet_demo/train_resnet50.py
```

2) a new job should appear



Job Name	Status
job-0	Running
tb-renset-demo-0	Running

3) The Tensorboard UI will refresh periodically

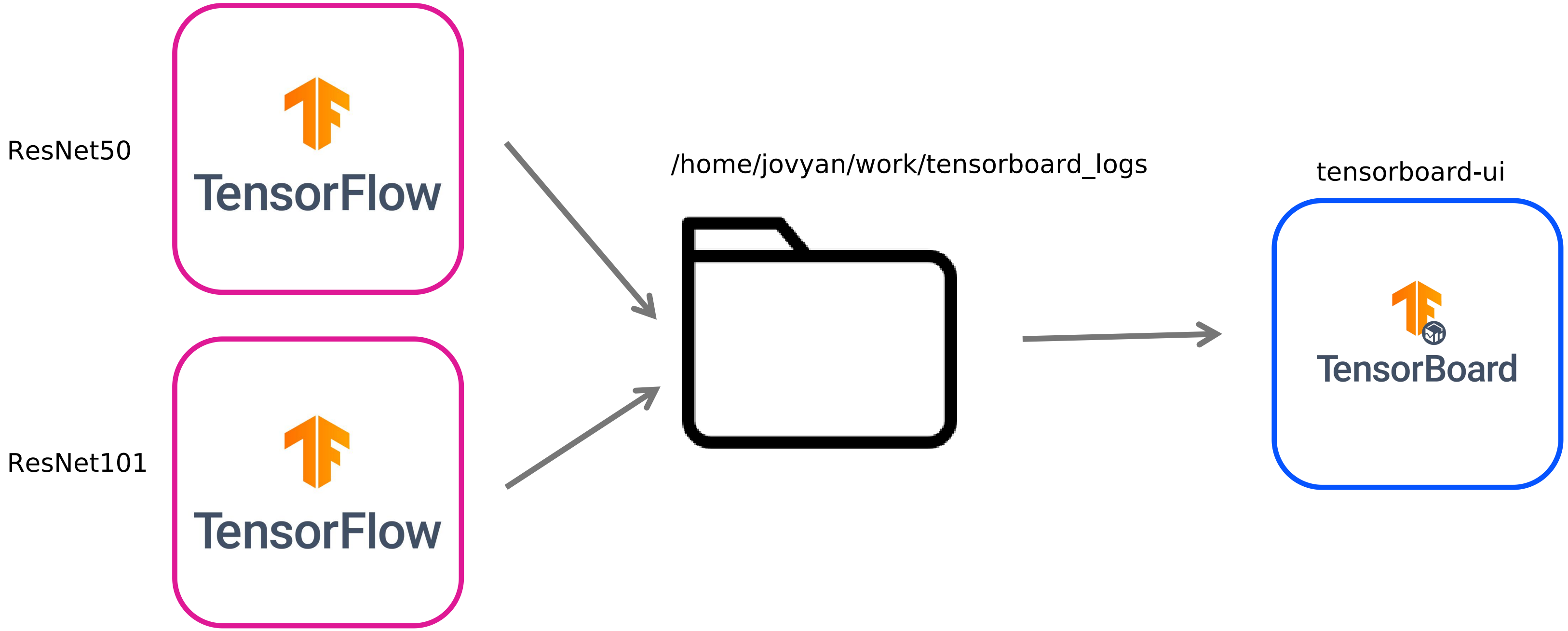


☐ Show data download links
☐ Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing 0.6
Horizontal Axis: STEP RELATIVE WALL
Runs
Write a regex to filter runs
☑ resnet50/20220817-122647/train
☑ resnet50/20220817-122647/validation
/home/jovyan/work/tensorboard_logs

epoch_loss
tag: epoch_loss

epoch_sparse_categorical_accuracy
tag: epoch_sparse_categorical_accuracy

Example job submission



Thank you!

info@run.ai | www.run.ai