DS-GA 1006 Capstone

Lab Session 2

Instructors: Julia Kempe
Mark Ho
Najoung Kim
Wenda Zhou

TA: Wenxin Zhang (wz2164@nyu.edu)

Lab 2 Session

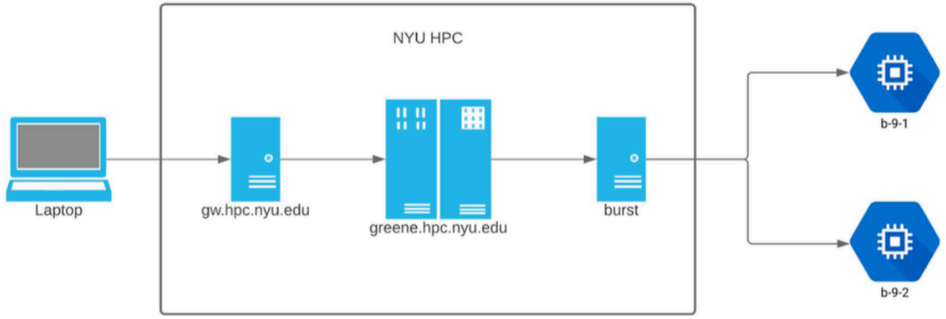
Topics

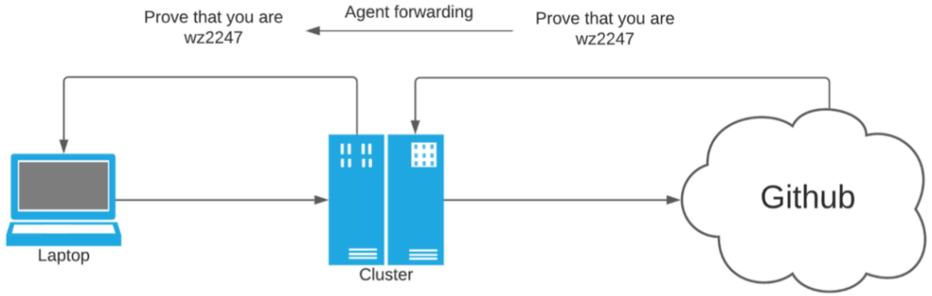
- Container Technology
- Singularity
- Singularity Overlays
- Singularity Instance

- Connect Singularity Instance on VSCode
- Run Basic Training Code
- Use Tensorboard to Visualize the Training Process

Review

- **SSH**
  - A method for secure remote login from one computer to another
- **Authentication Key Pairs**
  - Used for automating logins, single sign-on, and for authenticating hosts
  - Public Key & Private Key
- **SSH Agent & SSH Agent Forwarding**
  - SSH agent helps keep private key in memory so that you can log into the servers securely without a passphrase prompt
  - We can ask SSH to forward the authentication through agent forwarding





Homework Due Tomorrow

- **After finishing this homework, you have already know how to**
  - Configure your ~/.ssh/config file
  - Use key-based authentication
    - Connect your laptop to NYU HPC clusters
    - Connect Greene to Github
  - Request for GCP instance on burst login node
  - Connect GCP instance using VSCode remote SSH

- **Today, we will review how to use singularity on the GCP bursting server**

Container

- **What is Container?**
  - A lightweight, executable unit of software that packs up application code and dependencies such as binary code, libraries, and configuration files (within a self-sustainable image) for easy deployment across different computing environments

- **Why do we use Container?**
  - Reproduce your environment between Greene and GCP
  - Share the exact environment between team members
  - Help alleviate installation and portability challenges
  - Bypass inode limits on the cluster since it has a single physical file

Singularity

- A container platform specifically designed for use on HPC clusters
- Defined through the .sif file format
- Available on Greene and the GCP machines
- Can be accessed through command line
- The Nvidia pytorch docker image has been converted to a singularity image on GCP
  - IMAGE=/scratch/wz2247/singularity/images/pytorch_22.08-py3.sif

Run Singularity Containers

- Run Singularity Containers using “singularity exec” or “singularity shell”
  - singularity exec --no-home --cleanenv --nv --bind /scratch --bind $HOME/.ssh $IMAGE /bin/bash
  - singularity shell --no-home --cleanenv --nv --bind /scratch --bind $HOME/.ssh –shell=/bin/bash $IMAGE

- Use the --no-home argument and the --cleanenv argument, to increase the isolation between your container and the host
- Bind /scratch (--bind /scratch), and some subdirectory of $HOME (--bind $HOME/.ssh)
- On GPU machines, use the --nv flag to pass-through the GPUs

Overlays

- **Modify singularity containers**
  - The singularity image files (.sif) are immutable
  - Overlay a writable file system on the read-only container for the illusion of read_write access
  - Overlays can modify the base image and other overlays. The changes are kept separately from the base container image.
  - Mount the overlay when starting the container
  - Grab overlay from /scratch/work/public/overlay-fs-ext3

```
(base) wenxinzhang@10-19-49-243 ~ % ssh greene
Last login: Fri Sep 16 16:43:36 2022 from 10.47.6.5
[wz2164@log-3 ~]$ cd /scratch/work/public/overlay-fs-ext3/
[wz2164@log-3 overlay-fs-ext3]$ ls
overlay-0.1GB-25K.ext3.gz    overlay-0.75GB-300K.ext3.gz  overlay-1GB-400K.ext3.gz    overlay-2GB-100K.ext3.gz  overlay-4GB-300K.ext3.gz  overlay-5GB-3.2M.ext3.gz
overlay-0.25GB-100K.ext3.gz  overlay-10GB-400K.ext3.gz    overlay-2.4GB-1.5M.ext3.gz  overlay-2GB-1.3M.ext3.gz  overlay-50G-10M.ext3.gz   overlay-7.5GB-300K.ext3.gz
overlay-0.5GB-200K.ext3.gz   overlay-15GB-500K.ext3.gz    overlay-25GB-500K.ext3.gz   overlay-3GB-200K.ext3.gz  overlay-5GB-200K.ext3.gz
```

Example script to create overlay

- **Convert the base Nvidia PyTorch container to a customizable container with overlays**
  - Generally, overlays are best suited when you wish to logically modify the "image", which should contain your software dependencies.
  - It will usually be better to keep your own code, data and training results in the standard file system and access them through a bind point
  - Example scripts that create base overlay and package overlay (run on GCP instances)

Singularity Instances

- **Interactively work inside a container**
  - Use singularity instances to work interactively with writable overlays
  - Singularity instances commands
    - “singularity instance start” : need to specify all options for the container (e.g. bind paths, overlays). Example: “singularity instance start --containall $IMAGE myinstance”
    - “singularity instance list” : lists the set of all container instances that currently running
    - “singularity instance stop” : stop the instance with the given name
  - Run a shell in your given instance using ”singularity shell instance://myinstance”

```
[wz2164@b-2-1 ~]$ singularity instance start --containall /scratch/wz2247/singularity/images/pytorch_22.08-py3.sif myinstance
INFO:    instance started successfully
[wz2164@b-2-1 ~]$ singularity instance list
INSTANCE NAME    PID        IP    IMAGE
myinstance       1146808          /scratch/wz2247/singularity/images/pytorch_22.08-py3.sif
[wz2164@b-2-1 ~]$ singularity instance stop myinstance
INFO:    Stopping myinstance instance of /scratch/wz2247/singularity/images/pytorch_22.08-py3.sif (PID=1146808)
```

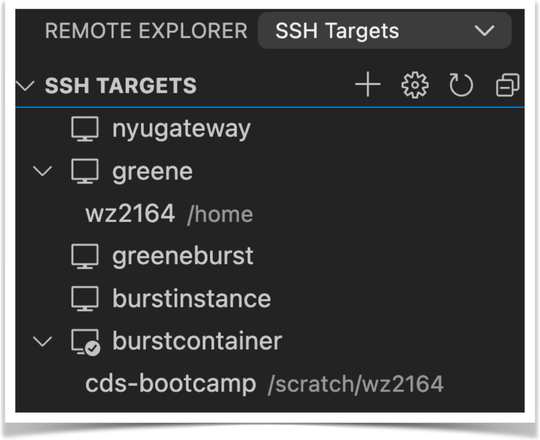
  - Example script that starts the singularity instance with recommended binds and parameters

Connect VSCode

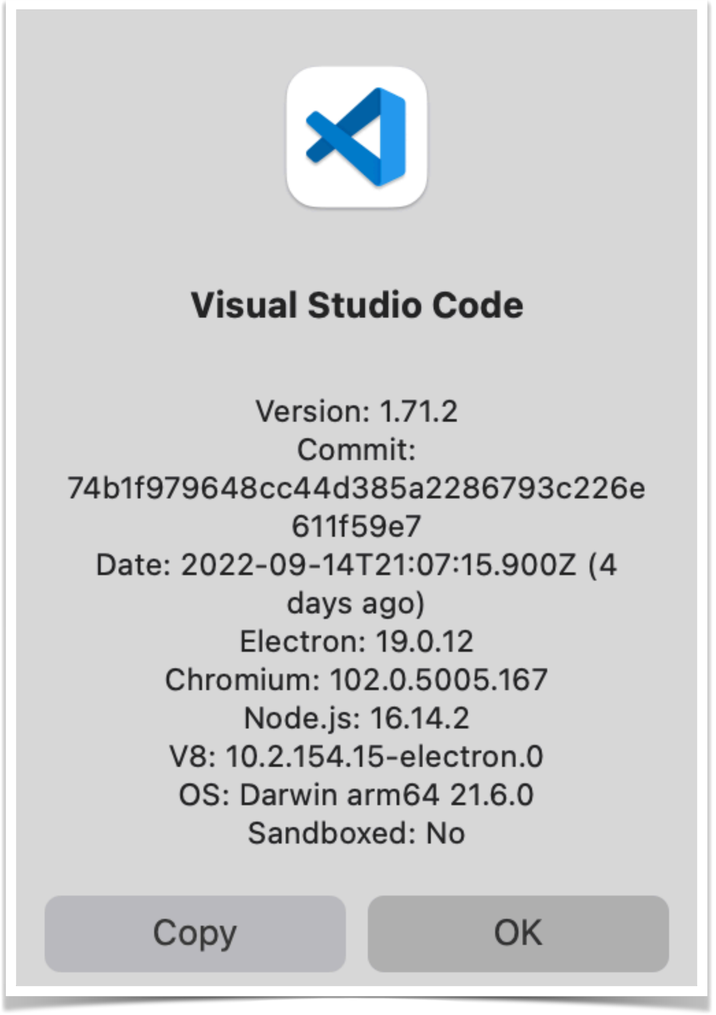
- **Use VSCode to connect to the singularity image**
  - Access your environment in an ergonomic fashion
  - Enable features like auto-completion, tests, debugging, etc.
  - Running a SSH server inside the instance is somewhat complex and prone to errors
    - Instead, make use of RemoteCommand option to drop our sessions directly to the container
    - With everything setup correctly, we can directly running inside the container

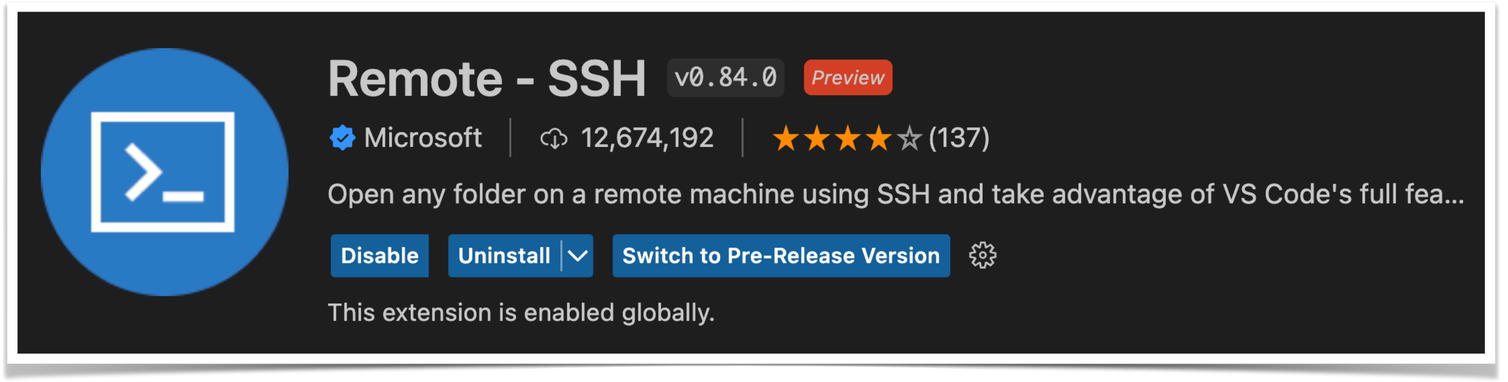
```
Host burstinstance burstcontainer
    User wz2164
    HostName b-9-5
    ForwardAgent yes
    ProxyJump greeneburst
    PasswordAuthentication yes
    ChallengeResponseAuthentication no
    StrictHostKeyChecking=No
    # UserHostsKnownFile=/dev/null


Host burstcontainer
    RemoteCommand singularity shell --containall --shell='/bin/bash' instance://mycontainer
    RequestTTY true
```

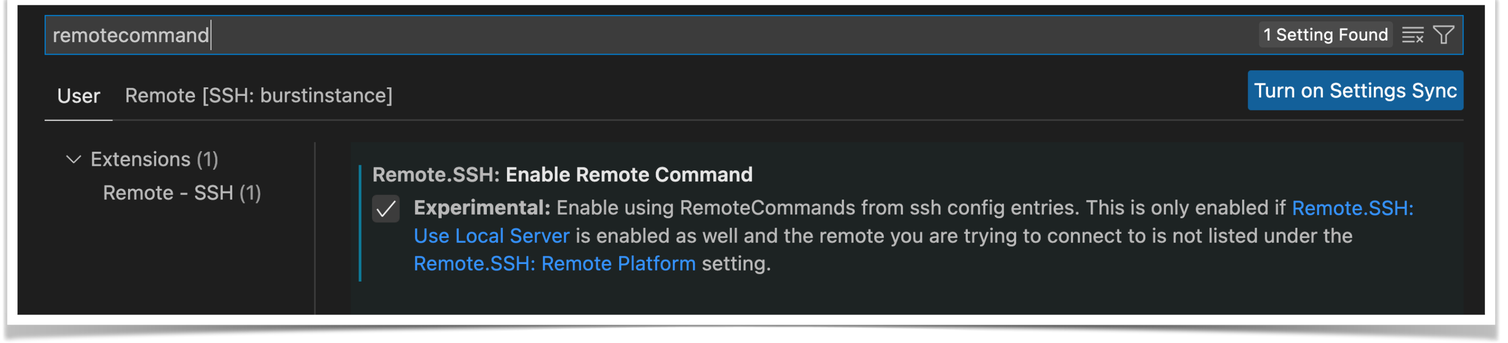



```
Singularity> ls /places365/
categories_places365.txt  places365_train_standard.txt
data_256_standard         places365_val.txt
places365_test.txt        val_256
Singularity>
```





- If you cannot connect the container directly from your laptop
  - Check VSCode version
  - Check Remote SSH version
  - Check RemoteCommand setting enabled



Start Singularity instance

- Drop our sessions directly to the container on VSCode

- Open the working directory /scratch/$NetID/cds-bootcamp/lecture2
  - ./script/create_base_overlay.sh
  - ./script/create_package_overlay.sh
  - ./start_singularity_instance.sh
  - singularity instance list

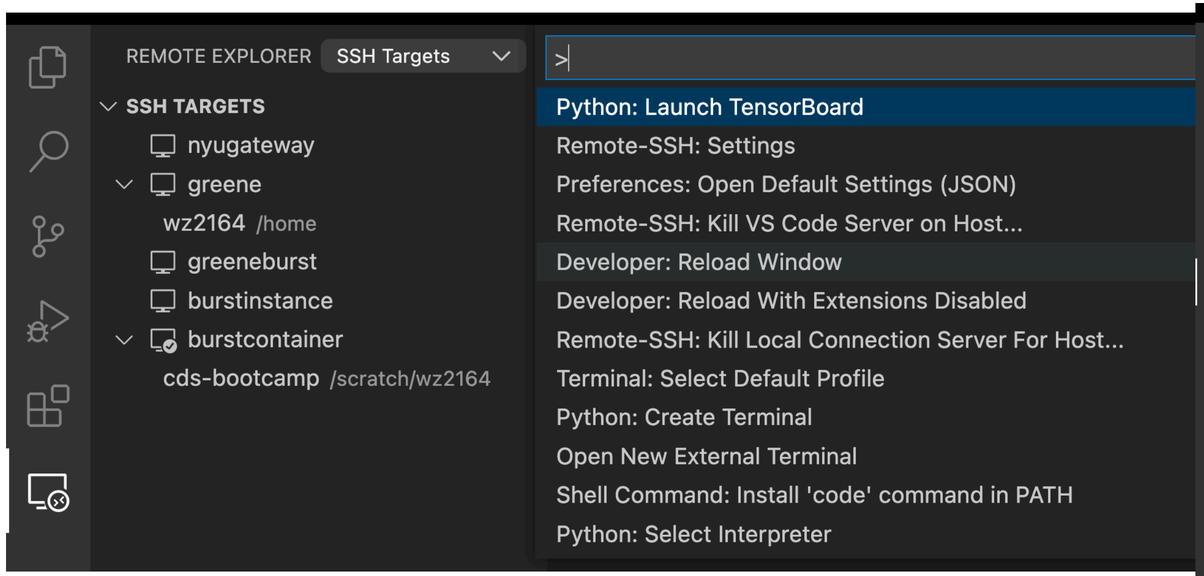
```
bash-4.4$ ./start_singularity_instance.sh
Temporary overlay not found, automatically creating a new one.
INFO:    instance started successfully
```

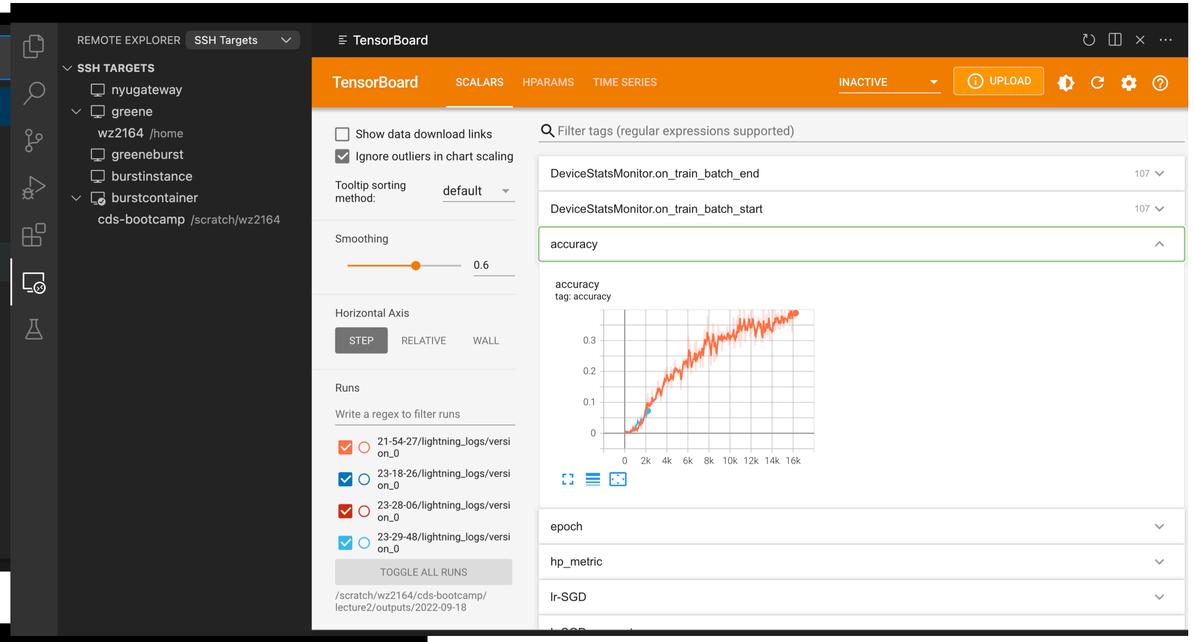
```
bash-4.4$ singularity instance list
INSTANCE NAME    PID       IP    IMAGE
mycontainer      814453          /scratch/wz2247/singularity/images/pytorch_22.08-py3.sif
```

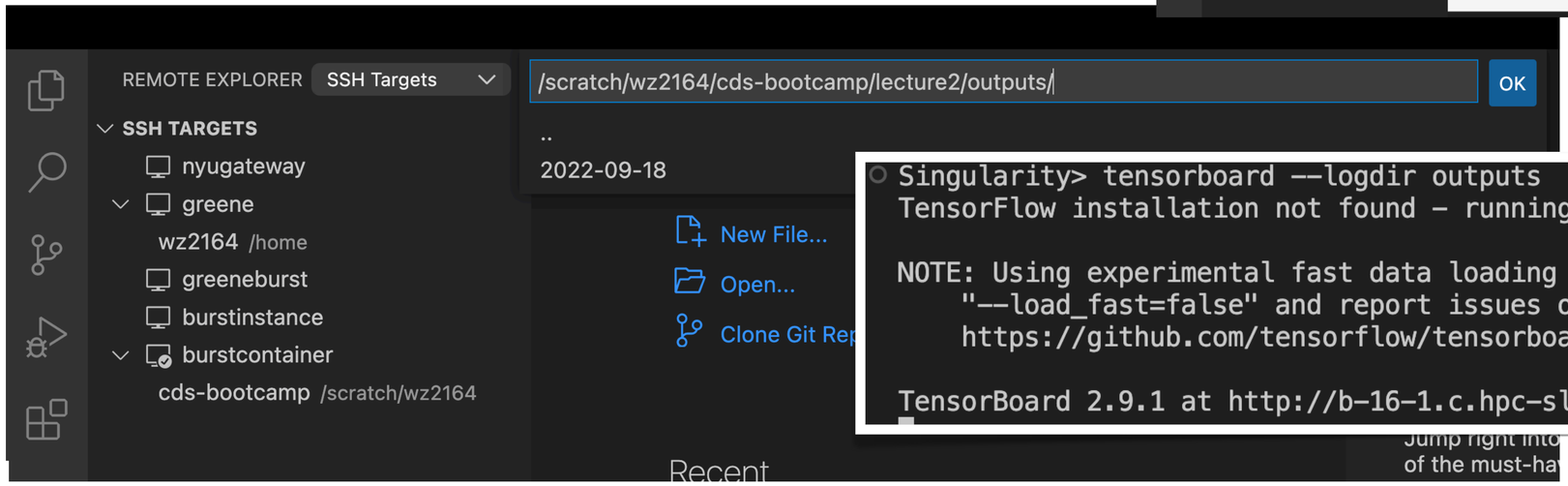
- Connet to the Singularity instance in the terminal
  - ssh burstcontainer
  - conda activate /ext3/conda/bootcamp
  - cd cds-bootcamp/lecture2
  - python -m bootcamp.train

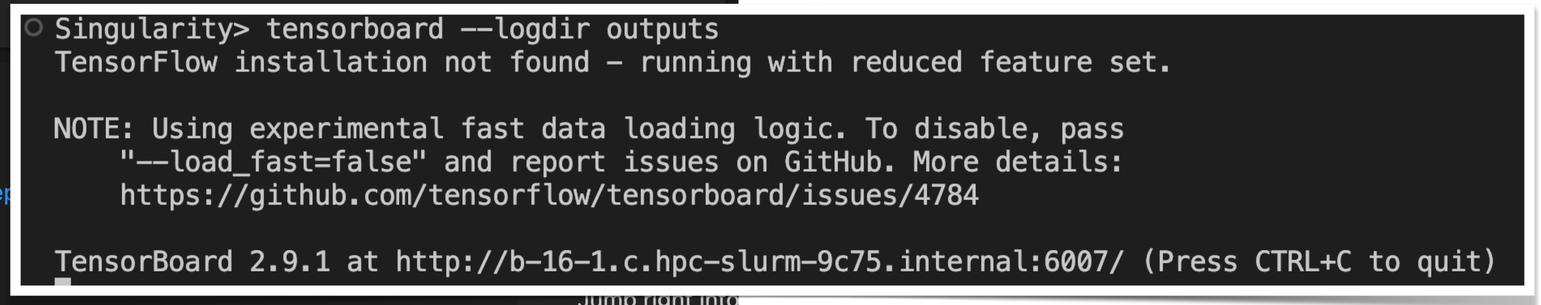
```
Singularity> conda activate /ext3/conda/bootcamp/
(/ext3/conda/bootcamp) Singularity> pwd
/home/wz2164
(/ext3/conda/bootcamp) Singularity> cd /scratch/wz2164/cds-bootcamp/lecture2/
(/ext3/conda/bootcamp) Singularity> ls
bootcamp           overlay-packages.ext3  start_singularity.sh
examples           overlay-temp.ext3      start_singularity_instance.sh
outputs            scripts
overlay-base.ext3  setup.py
(/ext3/conda/bootcamp) Singularity> python -m bootcamp.train
```

Track the Process using Tensorboard









VSCode Testing