EE-559 Deep learning – High Performance Computing, Students' questions

---

A question is denoted by **Q**; the corresponding answer is denoted by **A**.

Docker Images

Q: Is it normal that creating an image with Docker takes a lot of memory (up to 48GB)? When downloaded, will it also occupy the same amount of space on my local computer?
A: The size of the image is "normal", as the image supplied by Nvidia is already 27 GB, and adding the Python requirements takes up the rest of the space. If you're interested, you can use the "dive" tool to inspect your image. You can find an introduction to the use of this tool on the following webpage.

Q: I was able to use the image created by one of my teammates – my computer does not have enough memory to create the image.
A: If you don't have enough resources/space on your computer, please ask a member of your team to build an image for you using your UID/GID.

Q: Can we use each other's Docker images?
A: No, they are user-specific.

Q: Can I build the Docker image without a user ID, and then share the Docker Image with the whole group?
A: You are strongly recommended NOT to do this. While this is technically possible, you are likely to face significant limitations in access during the process.

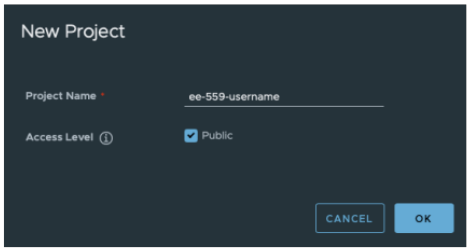
Q: If a Docker build is interrupted, does the rebuild process start from scratch?
A: No, it will restart from the point where it failed.

Q: I get the following error: "Error message: HTTPS proxy: connecting to nvrc.io:443: dial tcp: lookup nvrc.io: no such host". What should I do?
A: It's likely that the image name is incorrect – it should start with 'nvrc.io/...' rather than 'nvcr.io/...'.

Q: I got the following error when trying to build the image with Docker: "error during connect: Head "http://%2F%2F.%2Fpipe%2Fdocker DesktopLinuxEngine/_ping": open //./pipe/dockerDesktopLinuxEngine: The system cannot find the file specified". What should I do?
A: You need to enable virtualization on Windows (tutorial here). Then use the command "wsl --install".

Q: I got the following error when trying to build the image with Docker: "permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Head "http://%2Fvar%2Frun%2Fdocker.sock/_ping": dial unix /var/run/docker.sock: connect: permission denied". What should I do ?
A: You need to have Docker running on your PC to be able to interact with Docker.

Q: I got the following error:
"Error: Failed to copy: failed to do request: Put "https://registry.rcp.epfl.ch/...": write tcp 192.168.65.3:46290->192.168.65.1:3128: use of closed network connection." What is the problem ?
A: To fix the issue, make sure that your image is Public in the Harbor.


New Project

Project Name *: ee-559-username

Access Level: [✓ Public]

Q: I forgot a dependency in my Docker image. Do I need to reinstall everything?
A: No, Docker will start new installations from the first change in the Dockerfile, leaving the previous parts unchanged. This is why it is important to have the requirements.txt at the bottom of your Dockerfile – this means that the heaviest installation (which is coming from NVIDIA CUDA) will be taken as is.

Q: How do I update the requirements?
A:

1. Add the desired packages to the `requirements.txt` file. (Good practice is to always specify the version of the package you want, for reasons of reproducibility.)
2. Build your image (`docker build ...`). If you want to understand how Dockerfile layer works, you can find more details and examples on this website.
3. Push your image to the registry.
4. You can now use the new version of your image.

Q: What can be done once an image is built and transferred to the registry? Can we directly use the image from the registry (without relying on the local image)? Can others access the image and avoid the slow download process?
A: You can use the built image when submitting jobs on the HPC cluster. In that case, the image will be pulled from the registry rather than your local machine. Unless you are using base images (such as nvcr.io/nvidia/ai-workbench/pytorch:1.0.6), you cannot use images built by other users to run your job. Although the images in the registry are publicly

accessible, they are user-specific, as each image is built with individual user parameters (e.g., UID, GROUP_ID, etc.).

Running jobs and managing cluster storage

Q: I have a conflict between Python package installation in Docker Image and PVC mounting – I cannot use packages from my requirements.txt. What should I do?

A: We begin by explaining the cause of the error (included for completeness), followed by two suggested solutions. If you're only interested in the solutions, feel free to skip the explanation.

Cause: When a user builds a custom Docker image, they typically install all required Python packages listed in a requirements.txt file. By default, Python installs these packages in the following path: /home/<username>/.local/lib/python<version>/site-packages/

Consider the following example command used to submit a job to the cluster:

```
runai submit \
  --image registry.rcp.epfl.ch/ee-559-<username>/my-toolbox:v0.1 \
  --pvc home:${HOME} \
  -e HOME=${HOME} \
  --interactive \
  -g 1 \
  --attach
```

In this example, the persistent volume claim (PVC) named home is mounted into /home/<username>. This means that the contents of /home/<username> as defined in the Docker image (including the installed Python packages) are overwritten by the PVC mount. As a result, the Python packages installed during the image build become unavailable when running the job.

This has the following consequences:

- The Python packages installed in the Docker image are not accessible at runtime.
- This can lead to runtime errors due to missing dependencies.
- The behavior of the environment depends on the contents of the PVC, not the image, which breaks reproducibility.

Solution 1: Mount the PVC to a Different Path
Modify the mount point of the PVC to avoid overwriting /home/<username>:

```
runai submit \
  --image registry.rcp.epfl.ch/ee-559-<username>/my-toolbox:v0.1 \
  --pvc home:/pvc/home \
  -e HOME=/home/<username> \
  --interactive \
  -g 1 \
  --attach
```

This ensures that the packages installed in the image remain available at runtime.

Solution 2: Accept Mounting to /home/<username> *with Trade-Off*
It is also possible to intentionally mount the PVC to /home/<username> in order to benefit from persistent Python package installations without needing to rebuild the image. For example: pip install <package>. This can simplify usage of the cluster during development. However, this approach compromises reproducibility since the dependencies are now located on the PVC instead of being baked into the image.

Recommendation: To ensure reproducibility, prefer Solution 1.

Q: Our group uses GitHub – how should we make use of the shared storage?
A: Share your code on GitHub, and clone it to your home. Home is your working directory, so it should also include the dataset and the model's checkpoints, both of which will probably be too big to be shared via GitHub. This means that you would need to add models' checkpoints and the dataset folder to your .gitignore. To share the checkpoints and dataset with your teammates, copy them from your home to your group's shared storage.

Q: Can I run jobs from the shared folder?
A: No, you can only submit jobs from your home directory, which acts as your active workspace. This is why you need to "mount" your home directory (or a subfolder within it) into the container when submitting a job. Think of the shared folder as a convenient space for exchanging files among group members, rather than as a location from which you can directly run jobs.

Q: Can I access the shared group folder via something else than the terminal?
A: No, you need to use the terminal commands.

Q: I receive the following error: "unknown flag: --dataset_path Usage: runai submit [flags] -- [COMMAND] [args...] [options]". How to fix it?
A: Make sure that you use a single command line when launching your job.