Engineering Practice Tutorial #1 (10.21.2022)

October 31, 2022

Engineering Practice Tutorial

In this tutorial, we are going to cover management strategies in the lab

1. Components of paper

Research paper consists of three components

- **Code**
  - Managed by github
- **Table & Graph**
  - Research results
- **Data**

2. Reproducible Research

In user's point of view, they want to utilize or reproduce the results on the paper. Therefore, need to manage packages, code, and data.

- **Managing pacakge dependecies**
  - pip package
    - managed by 'requirements.txt' file
    - pip freeze -r
  - poetry package
  - docker image
- **Managing code**
  - github
    - snapshots of code
- **Managing data**
  - Publicly
    - Cloud services (S3)
  - Privately (Locally)
    - NFS
    - MINIO

3. GPU management

`SLURM` is used for GPU clustering and allocation

- `SLURM` - open source, fault-tolerant, and highly scalable cluster management and job scheduling system for large and small Linux clusters
- Features
  - Able to use bash scripts to initiate training on multi-GPU clusters
  - Able to assigning priorities over tasks (using multilevel queue)
    - High
    - Standard
    - Low
    - Intermediate
      - Debugging or checking whether scripts are running appropriately
- e.g.
  - bash commands

        * using only local desktop GPUs
    - qsub bash commands
        * using server and local GPUs (GPU clusters)
        * running scripts on local desktop will use server and local desktop GPUs
        * no need to directly executing scripts on server

4. Data management

- NFS (Network File System)
    - `/data`, `/home` directories are NFS
    - Locations
        * `/data`
            · large files w/o backup
        * `/home`
            · small files w/ backup
            · backup on hourly basis
            · user accounts are located under /home directory. Therefore, it would be a good practice to save extremely large models and data under `/data`
- Local
    - `/scratch`
        * for high frequency files (files frequently used such as caches)

5. Experiments management

- **Parameter Management**
    - Framework (easiest, fastest)
        * comet
        * wandb
        * tensorboard
    - Files
        * JSON
        * YAML
    - CMD output stdout or file
        * bash -v
            · verbose : print each command to stdout before executing it
        * bash -x
            · xtrace : Similar to -v, but expands commands
- **Model Management**
    - One folder per experiemnt
        * contains all necessary files related to the experiment : tokenizers, parameters, models, and etc.
    - Every experiments should have own unique identifier
        * e.g.) `{Path}/experiment/{github code tag or hash ID}/{slurm_id}/trial_1`
- **Code Management**
    - Use git tag feature (or hash ID) as identifier for training experiment
        * e.g.) `{Path}/experiment/{github code tag or hash ID}/{slurm_id}/trial_2`
    - Manage remote repository on individual laptops
        * Hard to track down corressponding code to each experiment when there are some direct modifications on the server.
    - Git pull codes from remote repository on the server
        * Easier to find corresponding code to each experiment