A Quick Start Guide to Develop on a Remote Server

Computer Vision (CS308 Fall 2020)

Minghui Chen

September 10, 2020

Southern University of Science and Technology

Table of contents

1. How to connect to remote server
2. Necessary tool
3. Pytorch Tutorial

How to connect to remote server

SSH command

```
ssh username@host(IP address) −p PORT
```

Username & init passwd: SID. (change the password: passwd)

IP address: 10.20.69.78(class of 15-17), 10.20.22.150(class of 18-22).

Port: 10022.

Options:

-C: Compresses all data (including stdin, stdout, stderr, and data for forwarded X11 and TCP connections) for a faster transfer of data.

-q: Suppresses all errors and warnings

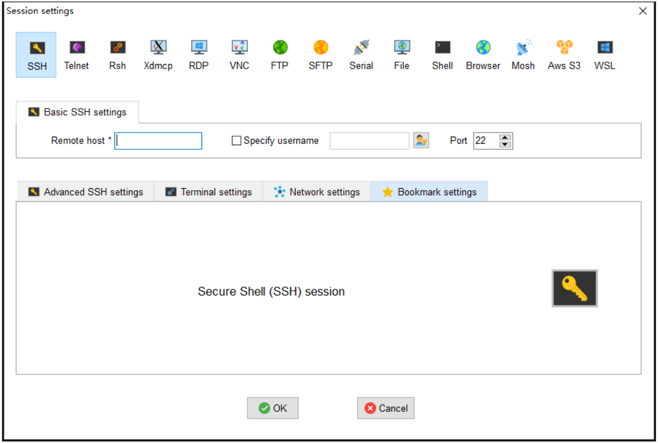
-v: Verbose mode. It echoes everything it is doing while establishing a connection. It is very useful in the debugging of connection failures

Privacy protection:

```
$ chmod 700 /disk/your_SID
```

MobaXterm

MobaXterm. Enhanced terminal for Windows with X11 server, tabbed SSH client, network tools and much more.



Remote host *

□ Specify username

Port 22

Secure Shell (SSH) session

Necessary tool

Linux basic commands

Linux basic commands

- **pwd**: Use the pwd command to find out the path of the current working directory (folder) you're in.
- **cd**: To navigate through the Linux files and directories, use the cd command. t requires either the full path or the name of the directory(You can use tab to complete the file name), depending on the current working directory that you're in. E.g. **cd ..** (with two dots) to move one directory up.
- **ls**: The ls command is used to view the contents of a directory. By default, this command will display the contents of your current working directory. E.g., **ls -R** will list all the files in the sub-directories as well. **ls -a** will show the hidden files. **ls -al** will list the files and directories with detailed information like the permissions, size, owner, etc.

Linux basic commands

Linux basic commands

- **cat**: (short for concatenate) is one of the most frequently used commands in Linux. It is used to list the contents of a file on the standard output (sdout). E.g., **cat filename1 filename2** > **filename3** joins two files (1 and 2) and stores the output of them in a new file (3).
- **cp**: Use the cp command to copy files from the current directory to a different directory. E.g., **cp -r dir1/ dir2/** copy the directory dir1 to the dir2 directory.
- **mv**: The primary use of the mv command is to move files, although it can also be used to rename files. E.g., **mv source destination** rename or move file(s) or directories.
- **mkdir**: Use mkdir command to make a new directory.

Linux basic commands

Linux basic commands

- **rm**: The rm command is used to delete directories and the contents within them. If you only want to delete the directory — as an alternative to **rmdir** — use **rm -r**. Additionally, **rm -rf *** can remove any file, but it is very dangerous!
- **grep**: It lets you search through all the text in a given file. E.g., grep blue notepad.txt will search for the word blue in the notepad file. Lines that contain the searched word will be displayed fully. **ps -ef** | **grep python** pipe the ps query result to grep to find the process that contains a specific string(e.g. python). Then you can use **kill -s 9 PID**(corresponding second string above) to kill program that have not been closed normally.

Linux basic commands

Linux basic commands

- **df**: Use df command to get a report on the system's disk space usage, shown in percentage and KBs. If you want to see the report in human readable format, type **df -h**.
- **du**: If you want to check how much space a file or a directory takes, the du (Disk Usage) command is the answer. E.g., **du -h –max-depth=1 2>/dev/null** to check the size of each folder in the current path, and ignore warning messages such as permission denied.
- **tar**: The tar command is the most used command to archive multiple files into a tarball — a common Linux file format that is similar to zip format, with compression being optional. E.g., **tar -zcvf filename.tar.gz /folder** gzip compression and **tar -zxvf filename.tar.gz** unzip files in current directory.

Linux basic commands

Linux basic commands

- **chmod**: chmod is another Linux command, used to change the read, write, and execute permissions of files and directories. As this command is rather complicated, you can read the full tutorial in order to execute it properly.
- **wget**: The Linux command line is super useful — you can even download files from the internet with the help of the wget command. To do so, simply type wget followed by the download link.
- **top**: As a terminal equivalent to Task Manager in Windows, the top command will display a list of running processes and how much CPU each process uses. **htop** is similar to the top command and can perform related operations (killing, renicing) on this process without inputting its PID. **htop** is an interactive process viewer in Linux system.

Linux basic commands

Linux basic commands

- **man**: Confused about the function of certain Linux commands? Don't worry, you can easily learn how to use them right from Linux's shell by using the man command. For instance, entering man tail will show the manual instruction of the tail command.
- **echo**: This command is used to move some data into a file. For example, if you want to add the text, "Hello, my name is John" into a file called name.txt, you would type **echo Hello, my name is John >> name.txt**
- **history**: Gives a list of all past commands typed in the current terminal session. You can use the arrow keys up and down to display historical commands.
- **clear**: Clear a command line screen/window for a fresh start.

Screen

screen command in Linux provides the ability to launch and use multiple shell sessions from a single ssh session.

Sometimes we need to run some tasks that take a long time to complete, such as system backup, ftp transfer, and so on. During this time, you cannot close the window or disconnect, otherwise the task will be killed. Screen can help us solve the above problem.

- **screen -ls** List all current sessions.
- **screen -S** `your_session_name` Create a new session called `your_session_name`.
- **ctrl** + **a** + **d** Detach, leave the current session temporarily, throw the current screen session (may contain multiple windows) to the background for execution, and return to the state when it has not entered the screen.
- **screen -r** `your_session_name` Resume offline screen jobs.
- **screen -S** `your_session_name` **-X quit** kill a screen or you can also enter the corresponding session and then **exit**.

VIM

Vim is a text editor developed from vi. Vim has two mode: Insert mode(Where you can just type like normal text editor. Press i for insert mode), Command mode(Where you give commands to the editor to get things done . Press ESC for command mode).

Most of them below are in command mode

- a - inserts text after the cursor.
- x - to delete the unwanted character.
- :wq - to save and exit
- :q! - to trash all changes
- u - to undo the last the command and U to undo the whole line.
- ctrl + r - to redo
- 0 - to move to the start of the line.
- $ - to the end of line
- G - to move you to the bottom of the file
- gg - to move you to the start of the file
- *k*G - 50G jump to line 50. *k* G jump to line *k*.

VIM

Other command mode hot key:

- h,j,k,l - left, down, up, right. 20j move down 20 lines. *k*j move down *k* lines.
- dw - move the cursor to the beginning of the word to delete that word and d2w deletes 2 words. Number can be changed for deleting the number of consecutive words like d3w
- dd - to delete the line and *k*dd to delete to *k* lines.
- yy - to copy the line and *k*yy to copy *k* lines.
- p - puts the previously deleted text after the cursor.
- / - backward search n to find the next occurrence and N to search in opposite direction.
- ? - forward search.
- v - starts visual mode for selecting the lines and you can perform operation on that like d delete.

VIM

Select all, copy and paste.

- delete all: esc, gg, dG
- copy all: esc, gg, ggyG
- select all highlight: esc, ggvG or ggVG
- paste: esc, p

Solve the problem that vim can only paste 50 lines:

Edit `~/.vimrc` in the current user's home directory (~) (if it does not exist, create a new file), add a line

```
:set viminfo='1000,<500
```

Anaconda

Anaconda Individual Edition is a free, easy-to-install package manager, environment manager, and Python distribution with a collection of 1,500+ open source packages with free community support.

Download command:

```
$ wget anaconda_download_url
```

Official: `https://repo.anaconda.com/archive/Anaconda3-2020.07-Linux-x86_64.sh`

SUSTech(Recommend!): `http://mirrors.sustc.us/anaconda/archive/Anaconda3-2020.07-Linux-x86_64.sh`

Install command:

```
$ sh ./Anaconda3-2020.07-Linux-x86_64.sh
$ source /disk/SID/.bashrc
```

Anaconda

Conda mirror setting:

```
$ conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/
$ conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main/
$ conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/pytorch/
$ conda config --set show_channel_urls yes
```

Pip mirror setting:

```
$ mkdir /disk/SID/.pip
$ vim /disk/SID/.pip/pip.conf
```

Then add following content:

```
[global]
index-url = https://pypi.tuna.tsinghua.edu.cn/simple
```

Create and switch environment:

```
$ conda create --name your_env_name python=3.8
$ conda activate your_env_name
```

Anaconda

Uninstall conda

```
$ rm -rf anaconda3(your anaconda dir)
$ vim /disk/SID/.bashrc
```

Use # at the end of the .bashrc file to comment out the previously added path (or delete it directly) and then save and exit:

```
export PATH=/disk/SID/anaconda3/bin:$PATH
```

View the current mirror and edit anaconda configuration file.

```
$ conda config --show-sources
$ vim /disk/SID/.condarc
```

Other domestic mirrors(Supplement the follow-up path by yourself):
`https://mirrors.sjtug.sjtu.edu.cn/anaconda/`

Pytorch

PyTorch is a machine learning library that shows that these two goals are in fact compatible: it provides an imperative and Pythonic programming style that supports code as a model, makes debugging easy and is consistent with other popular scientific computing libraries, while remaining efficient and supporting hardware accelerators such as GPUs.[1]

Install Pytorch:

```
$ conda install pytorch torchvision cudatoolkit=10.1
```

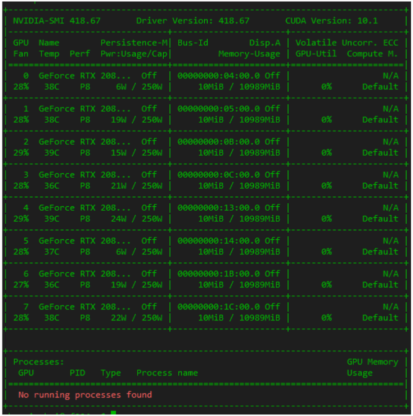
Verification. To check if your GPU driver and CUDA is enabled and accessible by PyTorch, run the following commands to return whether or not the CUDA driver is enabled:

```
$ python
>>> import torch
>>> torch.cuda.is_available()
```

GPU information command

Display GPU information:

```
$ nvidia -smi
```



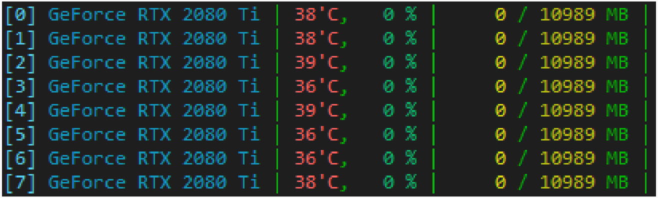
```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 418.67       Driver Version: 418.67       CUDA Version: 10.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce RTX 208...  Off  | 00000000:04:00.0 Off |                  N/A |
| 28%   38C    P8     6W / 250W |     10MiB / 10989MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  GeForce RTX 208...  Off  | 00000000:05:00.0 Off |                  N/A |
| 28%   38C    P8    19W / 250W |     10MiB / 10989MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   2  GeForce RTX 208...  Off  | 00000000:0B:00.0 Off |                  N/A |
| 29%   39C    P8    15W / 250W |     10MiB / 10989MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   3  GeForce RTX 208...  Off  | 00000000:0C:00.0 Off |                  N/A |
| 28%   36C    P8    21W / 250W |     10MiB / 10989MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   4  GeForce RTX 208...  Off  | 00000000:13:00.0 Off |                  N/A |
| 29%   39C    P8    24W / 250W |     10MiB / 10989MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   5  GeForce RTX 208...  Off  | 00000000:14:00.0 Off |                  N/A |
| 28%   37C    P8     6W / 250W |     10MiB / 10989MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   6  GeForce RTX 208...  Off  | 00000000:1B:00.0 Off |                  N/A |
| 27%   36C    P8    19W / 250W |     10MiB / 10989MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   7  GeForce RTX 208...  Off  | 00000000:1C:00.0 Off |                  N/A |
| 28%   38C    P8    22W / 250W |     10MiB / 10989MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

GPU information command

Display GPU information:

```
$ pip install gpustat
$ gpustat
```



```
[0] GeForce RTX 2080 Ti | 38'C,   0 % |     0 / 10989 MB |
[1] GeForce RTX 2080 Ti | 38'C,   0 % |     0 / 10989 MB |
[2] GeForce RTX 2080 Ti | 39'C,   0 % |     0 / 10989 MB |
[3] GeForce RTX 2080 Ti | 36'C,   0 % |     0 / 10989 MB |
[4] GeForce RTX 2080 Ti | 39'C,   0 % |     0 / 10989 MB |
[5] GeForce RTX 2080 Ti | 36'C,   0 % |     0 / 10989 MB |
[6] GeForce RTX 2080 Ti | 36'C,   0 % |     0 / 10989 MB |
[7] GeForce RTX 2080 Ti | 38'C,   0 % |     0 / 10989 MB |
```

VS Code

Install Remote-SSH extension and edit file `C://Users/Name/.ssh.config` with following lines:

```
    Host CS308_Lab
        HostName IP(e.g. 10.20.69.78)
        User SID
        Port 10022
```

Password-free login:

Generate a local key, the command is completed on the local computer:

```
    ssh−keygen −t rsa −b 4096
```

Upload the local public key(`id_ras.pub`) to the server and add it to the authorized keys file (`~/.ssh/authorized_keys`).

VS Code

Recommended extensions.

- Python: Linting, debugging, code navigation, code formatting, Jupyter notebook support, refactoring, variable explorer, test explorer, snippets, and more!
- Anaconda Extension Pack: Anaconda Extension Pack is a set of extensions that enhance the experience of Anaconda customers using VS code.
- Setting Sync: Synchronize settings, snippets, themes, file icons, launch, keybindings, workspaces and extensions across multiple machines using Github Gist.
- Code Runner: Run code snippet or code file for multiple languages.
- Code Spell Checker: Spelling checker for source code.
- Markdownlint: Markdown linting and style checking for VS code.
- LaTeX Workshop: LaTeX Workshop is an extension for Visual Studio Code, aiming to provide core features for LaTeX typesetting with Visual Studio Code.

Jupyter notebook

Use ssh to remotely access the server Jupyter Notebook.

0. Install jupyter notebook(using conda or pip) and setting login password: **jupyter notebook password**.
1. On the remote server, start the jupyter notebooks service: **jupyter notebook –no-browser –port=8855**(1024-65535 choose a number you like)
2. Start SSH in the local terminal: **ssh -N -f -L localhost:8888:localhost:8855 username@serverIP -p 10022**. (Among them: -N tells SSH that there is no command to be executed remotely; -f tells SSH to execute in the background; -L is the configuration of designated port forwarding, the remote port is 8855, and the local port number is 8888.)
3. Finally, open the browser and visit: `localhost:8888`

You can also directly create a file with the suffix .ipynb and open it with vscode. (The premise is that you have installed the Python extension in VS Code)

Pytorch Tutorial

Pytorch tutorail

Practice: Training a classifier.

Follow the tutorial(link) to experience training a neural network on GPU and run all the codes correctly.

Official tutorial: `https://pytorch.org/tutorials/beginner/deep_learning_60min_blitz.html`

Unofficial tutorial: `https://github.com/yunjey/pytorch-tutorial`

Questions?

References i

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala.
Pytorch: An imperative style, high-performance deep learning library.
In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 8024–8035, 2019.