연구실 서버컴퓨터 원격접속

MAC 기준입니다. 윈도우는 모름

> 연구실 IP 주소 : `166.104.246.53`
>
> SSH 포트번호 : `7001`
>
> 컴퓨터(ailib) 비밀번호 : `ai!0516`
>
> **포트번호는 개인이 임의로 설정해주면된다(약속된 포트번호제외)** 이 PDF에서는 예시로 설정했음
>
> - 로컬 포트번호 = 9999
> - 연구실 서버 포트번호 = 141414
> - 도커내부 포트번호 = 10000

참고사이트

SSH

https://eungbean.github.io/2019/03/22/jupyter-ssh/

http://programmingskills.net/archives/315

Docker

블로그1

블로그2

구글 클라우드

SSH 연결

터미널에서

```
ssh-keygen
```

입력 후 엔터누르면됨.

```
(base) minsuha@minsuhaui-MacBookAir ~ % ssh-keygen

Generating public/private rsa key pair.
Enter file in which to save the key (/Users/minsuha/.ssh/id_rsa):
/Users/minsuha/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/minsuha/.ssh/id_rsa.
Your public key has been saved in /Users/minsuha/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:CiMj6EpTRKhzrPCcK98O0fs5lJWouqut5bMIFsgcKfY minsuha@minsuhaui-MacBookAir.local
The key's randomart image is:
+---[RSA 3072]----+
|  ..             |
| .o              |
|o= .    . .      |
|@ *.   . o       |
|=X+E+. oS        |
|o.*+.+o.         |
|.=oo...          |
|*==o ...         |
|==B*o o.         |
+----[SHA256]-----+
```

중간에 `passphrase` 설정하라는게 나오는데 비밀번호 같은것으로 생각된다. 굳이 안해도 접속 가능함. 그리고 서버로 접속

```
ssh -L <로컬 포트번호>:localhost:<연구실 서버 포트번호> ailab@166.104.246.53 -p7001

# 원격포트번호 다른 사람들과 안겹치게 설정하기! 예를들면
ssh -L 8880:localhost:9999 ailab@166.104.246.53 -p7001
```

중간에 랩실 컴퓨터 비밀번호 입력

```
(base) minsuha@minsuhaui-MacBookAir ~ % ssh ailab@166.104.246.53 -p7001
The authenticity of host '[166.104.246.53]:7001 ([166.104.246.53]:7001)' can't be established.
ECDSA key fingerprint is SHA256:a/0NhGZE/m+yl9qs3T9TD9iA1iOl6pcaN7MnxJIm7ao.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[166.104.246.53]:7001' (ECDSA) to the list of known hosts.
ailab@166.104.246.53's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.8.0-63-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

61 updates can be applied immediately.
3 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable


20 updates could not be installed automatically. For more details,
see /var/log/unattended-upgrades/unattended-upgrades.log
Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Thu Jul 22 17:38:08 2021 from 166.104.246.54
ailab@ailab:~$ ls
anaconda3  cuda_11.2.2_460.32.03_linux.run  Desktop  docker  Documents  Downloads  g  memo  msha  Music  NVIDIA_CUDA-11.2_Samples  Pictures  Public  snap  Templates  Videos
```

Docker

conda create처럼 하나의 컴퓨터 안에서 독립적인 프로그래밍 환경을 만들어쓸 수 있음. conda create와 다른 점은

- 여러명이동시에 하나의 컴퓨터에 접근 가능하다고함
- 물론 컴퓨터 한대로 여러개의 notebook을 틀어놓고 실행할 수 있지만 하나가 무거운 작업을 하는 중이면 나머지는 거의 자원을 쓰지못해서 속도가 느려짐. 아마도 Docker는 가상환경(컨테이너)마다 자원을 분배해서 여러명이서도 비슷한 스펙을 쓸 수 있도록 해주는 것 같음

Docker에대한 간략한 설명

> git 허브에서 repo를 pull해서 코드를 쓰듯이 Deep learning 가상환경(pytorch, pandas 등등) 이미지를 pull해서 사용할 수 있다.

Docker 이미지

Docker는 환경(CUDA, CUDNN, pytorch)을 이미지형태로 push/pull 할 수 있다. 이미 최신버전 CUDA11, pytorch 이미지를 다운받아놨음.

```
ailab@ailab:~/Desktop/HA/share$ docker images
REPOSITORY          TAG                                 IMAGE ID       CREATED         SIZE
nvidia/cuda         9.0-base                            0bedd0dfd4cb   3 weeks ago     152MB
pytorch/pytorch     1.9.0-cuda11.1-cudnn8-devel         c35f519fe286   5 weeks ago     17.1GB
pytorch/pytorch     1.9.0-cuda11.1-cudnn8-runtime       12da3f8ec2ec   5 weeks ago     7.92GB
pytorch/pytorch     latest                              5ffed6c83695   4 months ago    7.25GB
ufoym/deepo         all-jupyter-py36-cu90               f8921d1fffaf   2 years ago     9.76GB
```

> pytorch `devel`과 `runtime`는 뭔지 모르겠어서 둘다 받아놨음
>
> **pytorch/pytorch:latest** 쓰면됨

Docker 가상환경(컨테이너)와 서버폴더 동기화

컨테이너는 가상환경이기 때문에 컨테이너에서 작업한 파일들은 컨테이너가 꺼지면 날라가지만 `서버(ailab)-컨테이너` 저장곤간을 묶어주면 컨테이너에서 작업한 것들이 서버에 저장되서 다음에도 쓸 수 있음. 그래서 컨테이너와 묶을 폴더하나를 만들어준다.

```
ailab@ailab:~/Desktop/HA$ ls -al
total 16
drwxrwxr-x  4 ailab ailab 4096  7월  24 17:52 .
drwxr-xr-x 10 ailab ailab 4096  7월  23 16:27 ..
drwxrwxr-x  2 ailab ailab 4096  7월  15 17:48 .ipynb_checkpoints
drwxrwxr-x  2 ailab ailab 4096  7월  24 17:52 share
```

- 저는 `~/Desktop/HA` 안에 `share`라는 폴더만듬

Docker 컨테이너 실행

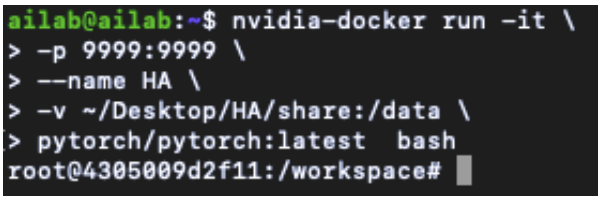
처음에는 컨테이너를 만들어주어야한다.

> -p : port
>
> --name : 컨테이너 이름
>
> -v : 서버-컨테이너 묶을 폴더(~/Docker로 고정)
>
> 마지막줄은 Docker 이미지 중 하나쓰면 됨

```
pytorch/pytorch:1.9.0-cuda11.1-cudnn8-devel
pytorch/pytorch:1.9.0-cuda11.1-cudnn8-runtime
pytorch/pytorch:latest
ufoym/deepo:latest
```

컨테이너 실행

```
nvidia-docker run -it \
-p <연구실 서버 포트번호>:<도커 내부 포트번호>\
--name <컨테이너 이름> \
-v ~/Docker:/data \
pytorch/pytorch:latest  bash


(Example)
nvidia-docker run -it \
-p 141414:10000\
--name HA \
-v ~/Docker:/data \
pytorch/pytorch:latest  bash
```



그러면 이렇게 새로운 shell이 실행됨. 여기에서 jupyter notebook을 설치해준다.

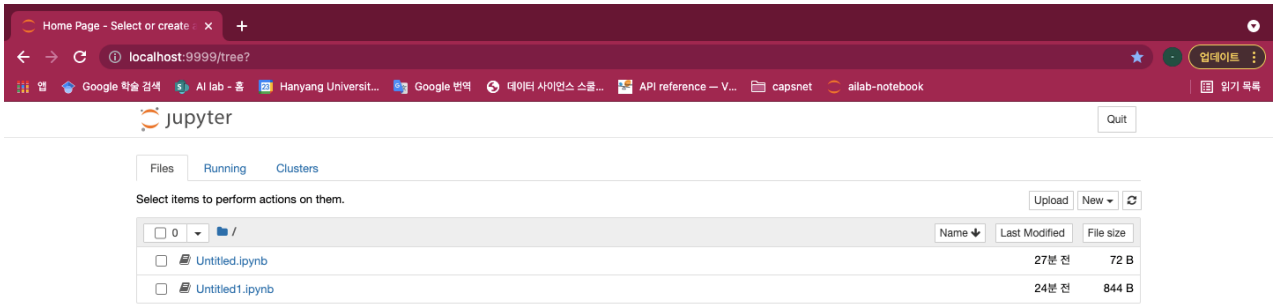
```
pip install ipykernel
pip install jupyter notebook
```

그리고 jupyter notebook 실행시켜준다.

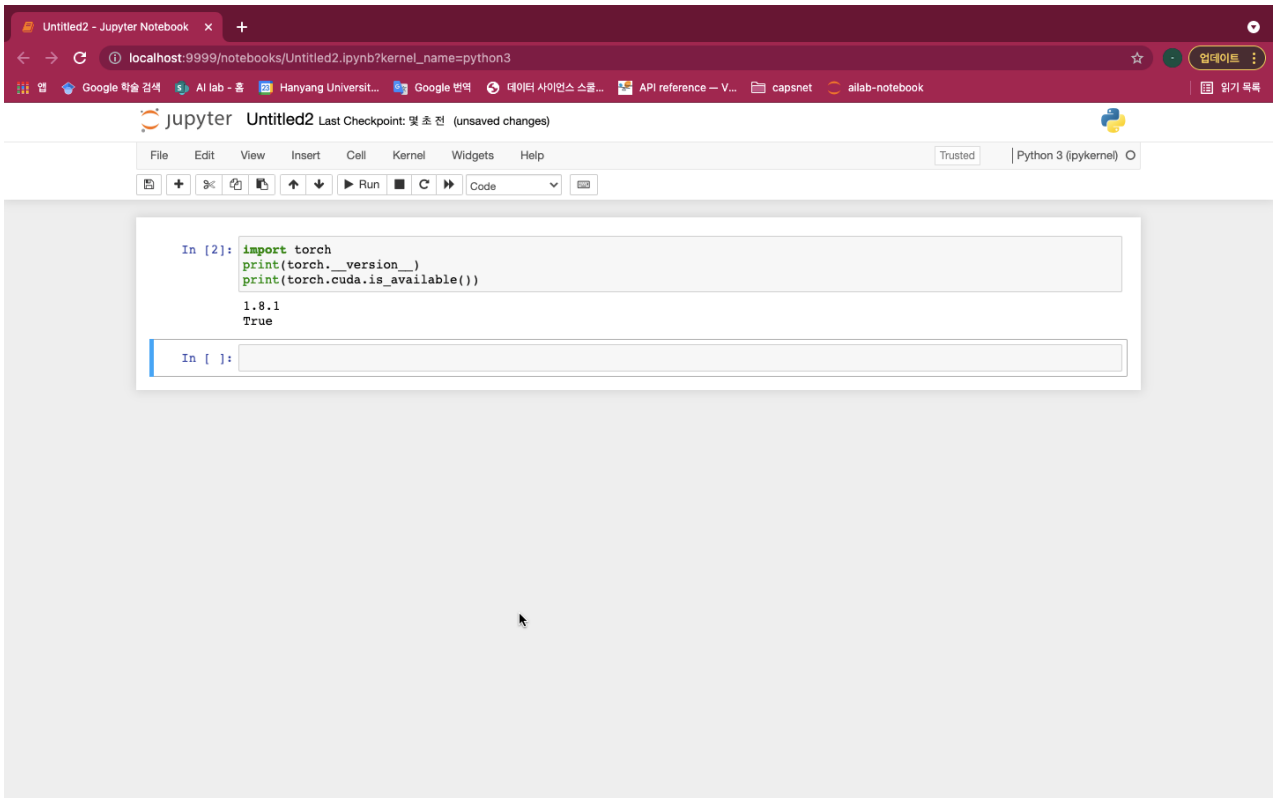
```
jupyter notebook --ip 0.0.0.0 --port <도커 내부 포트번호> --allow-root --
NotebookApp.token= --notebook-dir='/data'

(Example)
jupyter notebook --ip 0.0.0.0 --port 10000 --allow-root --NotebookApp.token= --
notebook-dir='/data'
```

그리고 chrome들어가서

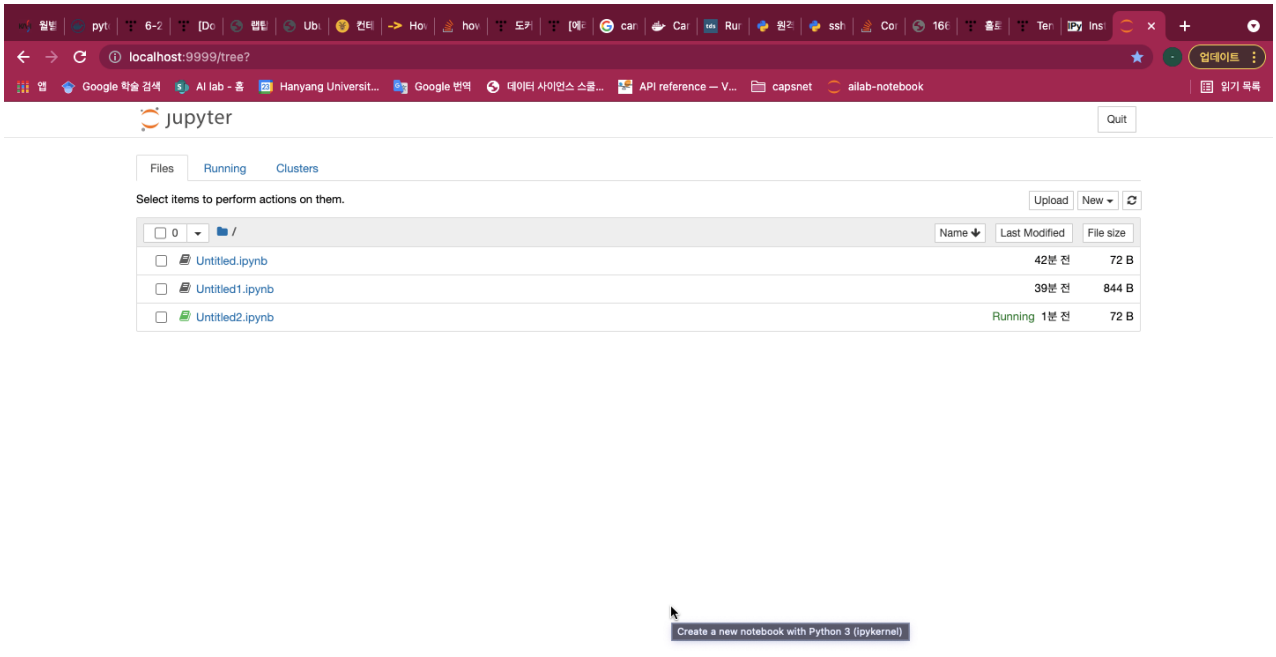
```
http://localhost:<로컬 포트번호>

(Example)
http://localhost:9999
```



를 입력해주면 끝. `nltk`같은 것도 그냥 `pip install something`으로 설치 해주고 notebook키면 된다.



torch gpu 사용가능




```
total 24
drwxrwxr-x 3 ailab ailab 4096  7월  25 14:09 .
drwxrwxr-x 4 ailab ailab 4096  7월  24 17:52 ..
drwxr-xr-x 2 root  root  4096  7월  25 14:09 .ipynb_checkpoints
-rw-r--r-- 1 root  root   844  7월  25 13:31 Untitled1.ipynb
-rw-r--r-- 1 root  root    72  7월  25 14:09 Untitled2.ipynb
-rw-r--r-- 1 root  root    72  7월  25 13:28 Untitled.ipynb
```

컨테이너에서 만든 파일들이 서버에 잘 묶이는것 확인 가능.

즐겨찾기 해놓고 컨테이너 킨다음 쓰면된다.

컨테이너 목록 보기

`la -al`과 비슷한 명령어

```
docker ps -a
```

```
ailab@ailab:~$ docker ps -a
CONTAINER ID   IMAGE                    COMMAND   CREATED         STATUS                          PORTS     NAMES
7771f8b073f6   pytorch/pytorch:latest   "bash"    3 minutes ago   Exited (0) About a minute ago             HA
```

종료 및 다시시작

작업을 끝내고 종료 시

```
exit
```

를 입력해주면 컨테이너 쉘에서 서버로 돌아와진다.

컨테이너를 다시 실행 시킬 시에는

```
docker start <컨테이너이름>
```

컨테이너를 start해준 뒤

```
docker attach <컨테이너이름>
```

attach를 해준다. *도커는 컨테이너를 실행시킨다는 표현보다 붙인다 라는 말을 쓴다고함*

Docker 명령어

자세한 내용

```
docker ps -a # 현재 실행중, 존재하는 컨테이너 목록 출력
docker rm <컨테이너이름 or 컨테이너ID> # 컨테이너 삭제
docker rmi <이미지이름 or 이미지ID> # 이미지 삭제
docker pull <이미지> # 이미지 pull
```

사용 포트번호

하민수

- 로컬 : any
- 서버 : [9999~10001]
- 컨테이너 : any

김지훈

- 로컬 : any
- 서버 : [10002~10004]
- 컨테이너 : any

이수아

- 로컬 : any
- 서버 : [10005~10007]
- 컨테이너 : any

빠스카

- 로컬 : any
- 서버 : [10008~10010]
- 컨테이너 : any