UBAI GUIDE BOOK

hs.hwang

2024-10-16

Table of contents

1 MAIN

UBAI Cluster .

Chapter .

UBAI Cluster

UBAI Cluster .

Chapter1.

UBAI .

Chapter2.

UBAI .

Chapter3.

, .

Chpater4. Python

Python .

Chpater5.

.

Chpater6.

.

UBAI GUIDE BOOK UBAI .

· AI .

서울시립대학교 | 도시과학빅데이터·AI연구원

2 UBAI Cluster

UBAI

· AI (UBAI) (HPC) .

Slurm AI , , (Job) .

Slurm

```
Slurm (Job Submit),
(Task Scheduling),
(Resource Management) Linux .
```

UBAI UBAI Cluster Slurm .

Slurm Slurm .

Visual Studio Code

```
Visual Studio Code( VScode) Microsoft .
MacOS, Linux, Windows ,
.
```

UBAI VScode .

VScode . VScode , VScode .

Partition

Slurm Partition .

Partition .

Partition .

Partition	# of Nodes	# of Cores/node	CPU	GPU/node	Memory/node	SSD	Note
gpu1	13	48	Intel Xeon Gold 6240R	RTX3090 (4EA)	768GB	2TB	*
edu1	5	48	Intel Xeon Gold 6240R	A10 (4EA)	768GB	2TB	*
cpu1	30	48	Intel Xeon Gold 6240R	None	768GB	2TB	*
gpu2	10	56	Intel Xeon Gold 6348R	A10 (8EA)	1024GB	2TB	*
gpu3	11	56	Intel Xeon Gold 6348R	A10 (4EA)	1024GB	2TB	*
gpu4	29	56	Intel Xeon Gold 6348R	A6000 (4EA)	1024GB	2TB	*
gpu5	6	64	Intel Xeon Platinum-8358	A6000 (4EA)	1024GB	2TB	*

※ UBAI 106 , 5,586 CPU , RTX3090 52 , A10 144 , A6000 140 .

Terminal Partition .

```
sinfo -o "%10P %5D %14F %4c %14G %N"
```

```
PARTITION  NODES  NODES(A/I/O/T)  CPUS  GRES           NODELIST
gpu1       13     10/3/0/13       48    gpu:rtx3090:4  n[001-013]
cpu1       35     16/19/0/35      48    (null)         n[014-048]
hgx        1      0/0/1/1         48    gpu:hgx:8      n050
gpu2       32     26/6/0/32       56    gpu:a10:4      n[051-070,073-080,083-086]
cpu2       14     14/0/0/14       56    (null)         n[087-100]
cpu3       6      4/2/0/6         64    (null)         n[101-106]
test       4      0/4/0/4         56    gpu:a10:4      n[071-072,081-082]
```

MaxJobs( ) 10, MaxSubmit( ) 20, MaxWall( ) 2 .

Partition	MaxJobs	MaxSubmit	MaxWall
*	10	20	2-00:00:00

AI , , .

( ) **AI** .

( ) The authors acknowledge the Urban Big data and AI Institute of the University of Seoul supercomputing resources (http://ubai.uos.ac.kr) made available for conducting the research reported in this paper.

3 Chapter1.

UBAI Cluster .

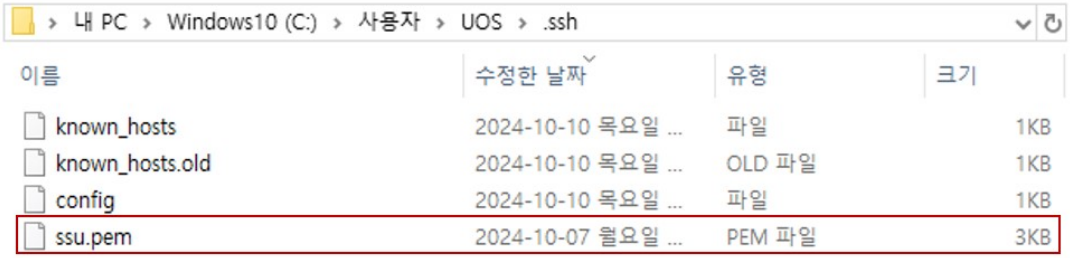
1.

UBAI .

(ubaisysadmin@uos.ac.kr) .

2.

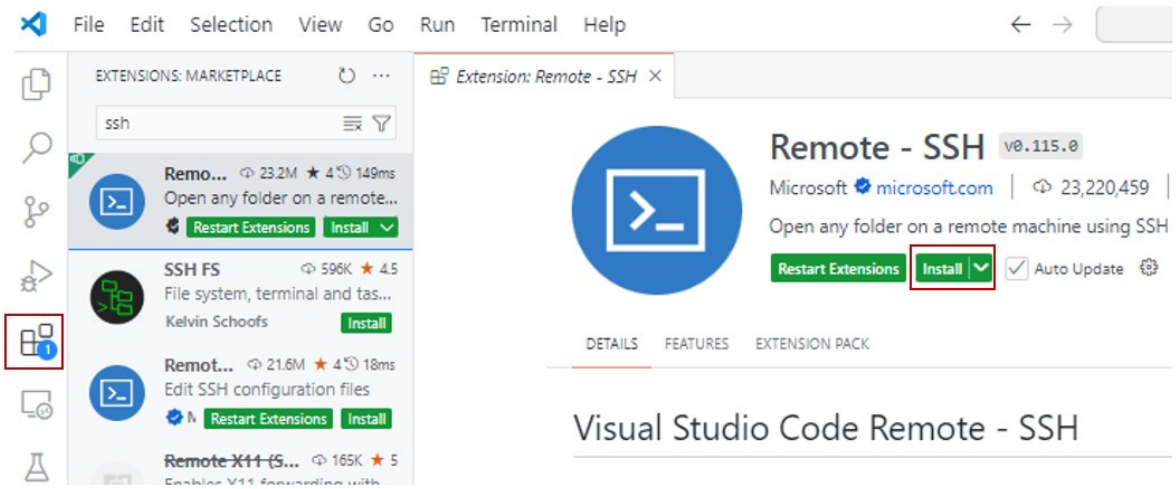
(ID.pem) `C:\User\{ }\.ssh\` .



※ *.ssh* , ***Chapter2.*** *.ssh* .

4 Chapter2.

1. Remote-SSH



VScode Remote-SSH . SSH .

SSH

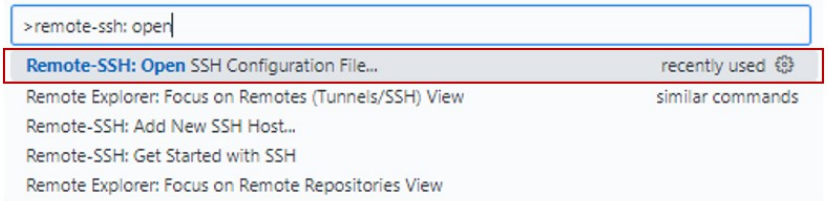
SSH
command .

SSH , VScode .

VScode Extension Remote-SSH .

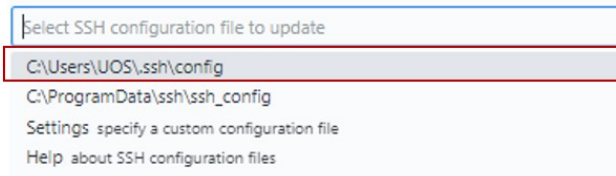
2. Config

1. Remote-SSH , (**CTRL + P**) (search) `>remote-ssh : open ssh`



   `configuration` .

※ *`.ssh`* , *`.ssh`* *config* . *`.ssh`* *config* .



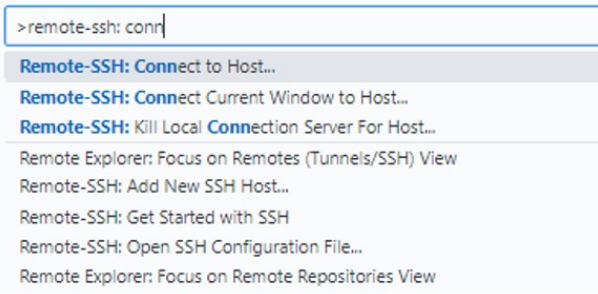
2. , `C:\Users\ \.ssh\config` .
3. `.ssh` config .

config

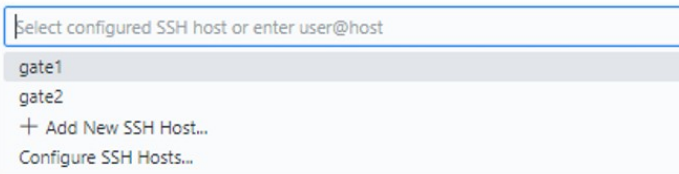
```
Host gate1
    HostName 172.16.10.36
    Port 22
    User   ID
    IdentityFile

Host gate2
    HostName 172.16.10.37
    Port 22
    User   ID
    IdentityFile
```

3. SSH

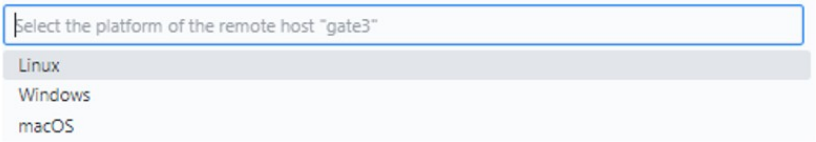
1. VScode (**CRTL+SHIFT+P**) .



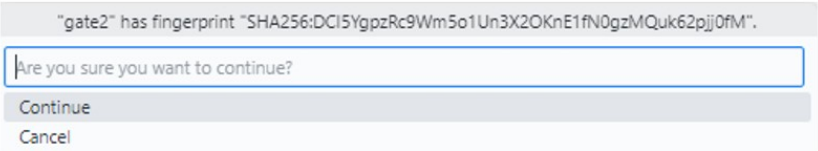
2. `> remote-ssh : connect to host` , .



3. gate1 gate2 . gate1 gate2 .

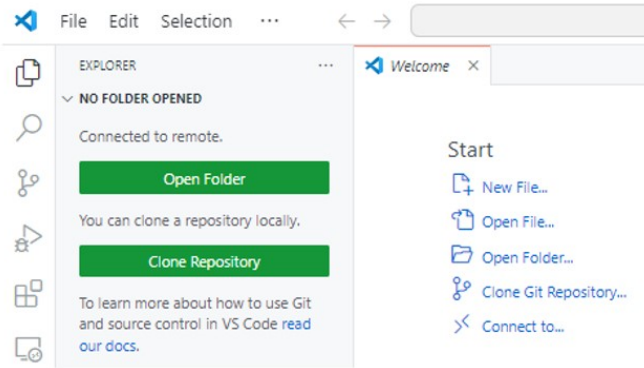


4. Linux .



5. Continue .

6. gate , gate SSH:gate .



7. , (Explorer) , Open Folder .
8. `/home1/{ ID}` , OK . .
9. , SSH !

5 Chapter3.

1. Linux

Environment Modules

```
Environment Modules
Unix/Linux                    .
Environment Modules       (PATH  )
              .

    (https://modules.sourceforge.net/)          .
```

1.1 Module avail

UBIA Cluster   .

```
(ubai) [ssu@gate1 ~]$ module avail

---------------------- /opt/ohpc/pub/modulefiles ----------------------
   CUDA/11.2.2             cmake/3.24.2                  cuda/11.2.2
   EasyBuild/4.9.1         compiler-rt/latest            cuda/11.3.1
   R/4.3.1                 compiler-rt/2023.1.0   (D)    cuda/11.4.4
   advisor/latest          compiler-rt32/latest          cuda/11.5.2
   advisor/2023.1.0 (D)    compiler-rt32/2023.1.0 (D)    cuda/11.6.2
   autotools               compiler/latest               cuda/11.7.1
   ccl/latest              compiler/2023.1.0      (D)    cuda/11.8.0
   ccl/2021.9.0     (D)    compiler32/latest             cuda/12.0.0
   clck/latest             compiler32/2023.1.0    (D)    cuda/12.1.1
   clck/2021.7.3    (D)    cuda/leejihun_cuda            cuda/12.2.1 (D)

  Where:
   D:  Default Module

If the avail list is too long consider trying:
```

```
"module --default avail" or "ml -d av" to just list the default modules.
"module overview" or "ml ov" to display the number of modules for each name.

Use "module spider" to find all possible modules and extensions.
Use "module keyword key1 key2 ..." to search for all possible modules matching any of the "ke
```

1.2 Module show

.

```
(ubai) [ssu@gate1 ~]$ module show cuda/11.2.2
-------------------------------------------------
   /opt/ohpc/pub/modulefiles/cuda/11.2.2:
-------------------------------------------------
whatis("Name: CUDA Collection")
whatis("Version: 11.2.2")
whatis("Category: cuda")
prepend_path("PATH","/opt/ohpc/pub/cuda/11.2.2/bin")
prepend_path("INCLUDE","/opt/ohpc/pub/cuda/11.2.2/include")
prepend_path("LD_LIBRARY_PATH","/opt/ohpc/pub/cuda/11.2.2/lib64")
family("cuda")
help([[
This module loads the CUDA

Version 11.2.2

]])
```

1.3 Module load

. .

```
(ubai) [ssu@gate1 ~]$ module load cuda/11.2.2
```

1.4 Module list

.

```
(ubai) [ssu@gate1 ~]$ module list

Currently Loaded Modules:
  1) cuda/11.2.2   2) dal/latest
```

1.5 Module rm

.

```
(ubai) [ssu@gate1 ~]$ module rm dal/latest

Removing dal version 2023.1.0
Use `module list` to view any remaining dependent modules.
```

1.6 Module purge

`module rm` . `module purge` `module list` module
.

```
(ubai) [ssu@gate1 ~]$ module purge

(ubai) [ssu@gate1 ~]$ module list
No modules loaded
```

2. Python

Python

`Python` , , ,
, .
, .

Anaconda Miniconda . UBAI Miniconda
.

Miniconda

`Anaconda`

.

.

`Miniconda` `Anaconda` .

2.1 Minicoda

Miniconda Miniconda .

1. terminal terminal . Miniconda .

   `wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh`

2. wget bash .

   `bash Miniconda3-latest-Linux-x86_64.sh`

3. Miniconda . Enter . 'yes' . Enter .

4. Enter .

5. , conda init . 'yes' enter .

6. .

7. , terminal `(base)[ ID@ _gate_number]` . (Explorer) miniconda .

2.2 Minicoda

Miniconda , Python .

Python (Package Dependencies) . Python .

1. terminal .

`conda create -n { _ } python={ _Python_ }`

ex. `conda create -n ubai python=3.11`

2. .

`conda activate { _ }`

ex. `conda activate ubai`

Jupyter notebook , . `pip install ipykernel jupyerlab` or `conda install ipykernel jupyterlab`

※ *Python* ***pip install*** .

6 Chapter4. Python

Python .

Shell **Python** . Shell **Jupyter Notebook** **, Python** .

.

1. BASH

Bash .

(job) , . (ubaisysadmin@uos.ac.kr) .

.

1.1

.

(job) , `filename.sh` . Shell `python_project.sh` .

`.sh` .

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --partition=gpu2
#SBATCH --cpus-per-task=56
#SBATCH --gres=gpu:4
#SBATCH --job-name=UBAIJOB
#SBATCH -o ./          /jupyter.%N.%j.out  # STDOUT
#SBATCH -e ./          /jupyter.%N.%j.err  # STDERR

echo "start at:" `date`
```

```
echo "node: $HOSTNAME"
echo "jobid: $SLURM_JOB_ID"

module unload CUDA/11.2.2
module load cuda/11.8.0

python cnn.py 12 256 'relu'
```

. .

STDOUT , STDERR . (directory)
.

.

`#SBATCH --nodes=1` , . nodes=1 .

`#SBATCH --partition=gpu4` Partition . Partition **UBAI Cluster** .

`#SBATCH --cpus-per-task=14` . n , 1 CPU/GPU
. #of Cores/node Partition . **UBAI Cluster** .

`#SBATCH --gres=gpu:1` GPU . CPU Partition . GPU
, .

`#SBATCH --job-name=UBAIJOB` .

`echo "start at:" 'date'` .

`echo "node: $HOSTNAME"` .

`echo "jobid: $SLURM_JOB_ID"` jobid .

`module ~` Linux . GPU , GPU (CPU Partition) .
Chapter3. module envrionment . .

`python cnn.py 12 256 'relu'` Python . `.py` . `cnn.py`
. Python sys `sys.argv` . sys .
`python {filename}.py` .

`.py` sys , .

- **sys.argv[0]:**
- **sys.argv[n]: (n .)**

`cnn.py` , `12 256 'relu'` sys.argv[1], sys.argv[2], sys.argv[3] .

cnn.py .

```
import sys
import tensorflow as tf
import keras
import time
import os

from tensorflow.python.keras import layers
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D

start = time.time()

img_rows = 28
img_cols = 28

(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()

input_shape = (img_rows, img_cols, 1)
x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
x_test = x_test.reshape(x_test.shape[0], img_rows, img_cols, 1)

x_train = x_train.astype('float32') / 255. #
x_test = x_test.astype('float32') / 255.  #

print('x_train shape:', x_train.shape)
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')

batch_size = int(sys.argv[2])
num_classes = 10
epochs = int(sys.argv[1])

y_train = keras.utils.to_categorical(y_train, num_classes) #
y_test = keras.utils.to_categorical(y_test, num_classes) #

model = Sequential()
model.add(Conv2D(32, kernel_size=(5, 5), strides=(1, 1), padding='same', activation='relu', i
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
```

```
model.add(Conv2D(64, (2, 2), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten()) # fully connected layer
#                   .
# conv2d            pooling            dense layer          feature map  input
model.add(Dense(1000, activation=sys.argv[3]))  # -> Dense Layer                    .
model.add(Dropout(0.5)) #
model.add(Dense(num_classes, activation='softmax'))
model.summary()

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
hist = model.fit(x_train, y_train, batch_size=batch_size, epochs=epochs, verbose=1, validati

score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])

end = time.time() - start

print(end)
```

1.2

, Python .

, terminal `sbatch` . (job) .

(job) ID .

```
sbatch filename.sh    # ex) sbatch python_project.sh
```

※ *cnn.py* ***pip install tensorflow && pip install numpy*** .

(job) , STDOUT OUT .

OUT , Partition (job) . terminal `squeue` , ID .

n001, n002 … , (*Resources, Priority*) .

, .

Partition Partition cpus-per-task, gpu Partition (job) .

STDOUT OUT .

```
jupyter.n013.206248.out ×
cnn_output > jupyter.n013.206248.out
1   start at: Tue Oct 8 09:47:10 KST 2024
2   node: n013
3   jobid: 206248
4
5   ------------------------- /opt/ohpc/pub/modulefiles -------------------------
6      CUDA/11.2.2           cuda/11.3.1    cuda/11.6.2    cuda/12.0.0
7      cuda/1eejihun_cuda    cuda/11.4.4    cuda/11.7.1    cuda/12.1.1
8      cuda/11.2.2           cuda/11.5.2    cuda/11.8.0    cuda/12.2.1 (D)
9
10     Where:
11      D:  Default Module
12
13  If the avail list is too long consider trying:
14
15  "module --default avail" or "ml -d av" to just list the default modules.
16  "module overview" or "ml ov" to display the number of modules for each name.
17
18  Use "module spider" to find all possible modules and extensions.
19  Use "module keyword key1 key2 ..." to search for all possible modules matching
20  any of the "keys".
21
22
23  Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
24
25     8192/11490434 [..............................] - ETA: 0s
26    16384/11490434 [..............................] - ETA: 48s
27    40960/11490434 [..............................] - ETA: 33s
```

2. Jupyter Notebook

Jupyter notebook .

(job) , . (ubaisysadmin@uos.ac.kr)

.

.

2.1

.

(job) , `filename.sh` . Shell `jupyter_notebook.sh` .

`.sh` .

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --partition=gpu4
#SBATCH --cpus-per-task=14
#SBATCH --gres=gpu:1
#SBATCH --job-name=UBAIJOB
```

```
#SBATCH -o ./          /jupyter.%N.%j.out  # STDOUT
#SBATCH -e ./          /jupyter.%N.%j.err  # STDERR

echo "start at:" `date`
echo "node: $HOSTNAME"
echo "jobid: $SLURM_JOB_ID"

module unload CUDA/11.2.2
module load cuda/11.8.0

python -m jupyter lab $HOME \
        --ip=0.0.0.0 \
        --no-browser
```

. .

STDOUT , STDERR . (directory) .

.

#SBATCH --nodes=1 , . nodes=1 .

#SBATCH --partition=gpu4 Partition . Partition UBAI Cluster .

#SBATCH --cpus-per-task=14 . n , 1 CPU/GPU . #of Cores/node Partition . UBAI Cluster .

#SBATCH --gres=gpu:1 GPU . CPU Partition . GPU , .

#SBATCH --job-name=UBAIJOB .

echo "start at:" 'date' .

echo "node: $HOSTNAME" .

echo "jobid: $SLURM_JOB_ID" jobid .

module ~ Linux . GPU , GPU (CPU Partition) . Chapter3. module envrionment . .

python -m jupyter lab $HOME \ --ip=0.0.0.0 \ --no-browse Jupyter notebook .

2.2

, Python .

, terminal `sbatch` . (job) .

(job) ID .

```
sbatch filename.sh    # ex) sbatch jupyter.sh
```

(job) , STDOUT OUT .

OUT , Partition (job) . terminal `squeue` , ID .

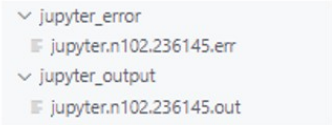
n001, n002 … , (*Resources, Priority*) .

, .

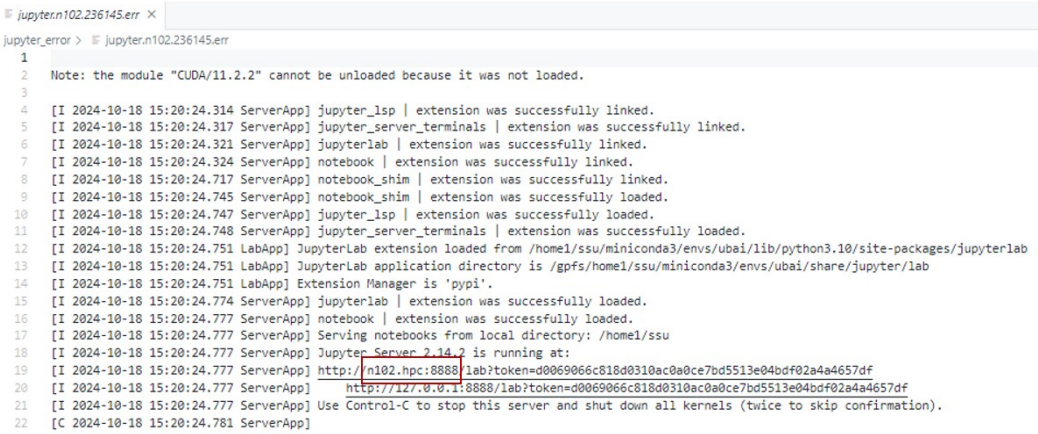
Partition Partition cpus-per-task, gpu Partition (job) .

ERR , Jupyter Notebook .

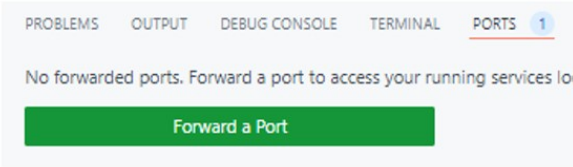
2.3



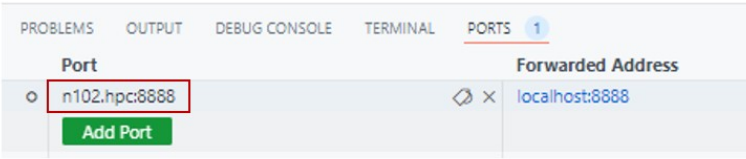
1. STDERR ERR .



2. .



3. VScode PORTS , Forward a Port .

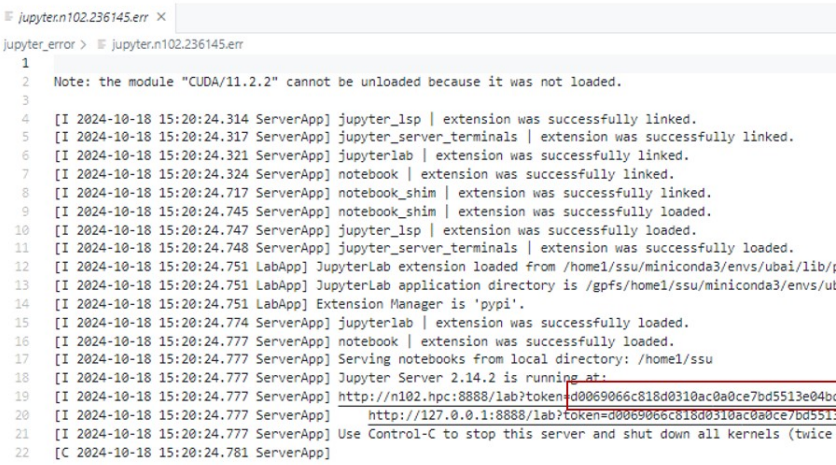


4. 2 Port , Open in Browser .
5. Jupyter .

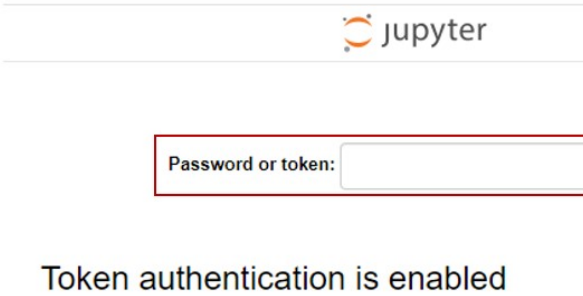
2.4 Jupyter Notebook

Jupyter , . Jupyter Notebook . .

1.



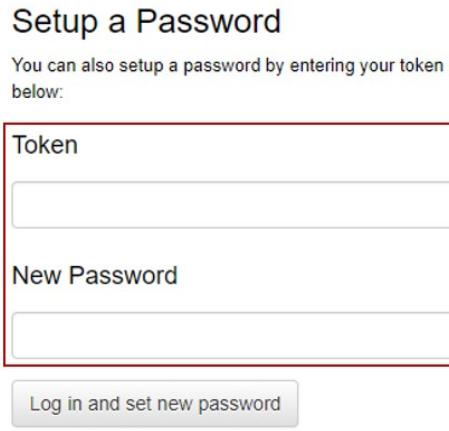
STDERR ERR Token .



Token . Token , Password .

1 , Token .

2.



Token , Token New Password Password .

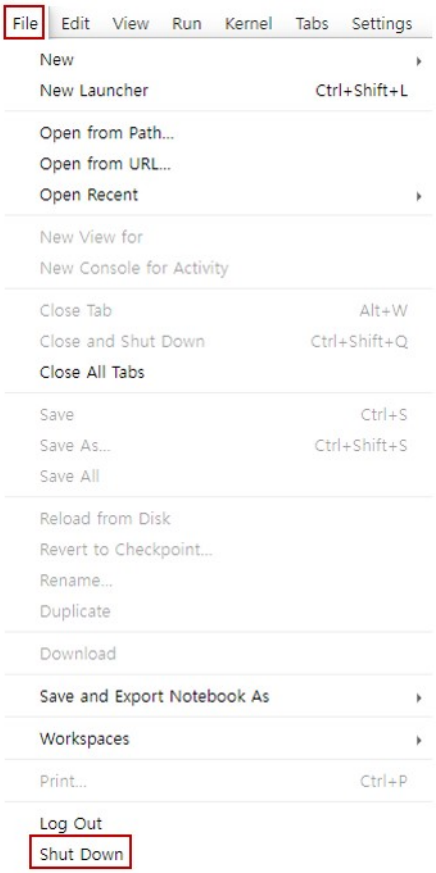
2 , Password .

Jupyter Notebook .

2.5 Jupyter Notebook

Jupyter Nobetook **File → Shut Down** , VScode terminal `scancel {job_ID}` . , job_ID job_ID .

(job) . .



7 Chapter5.

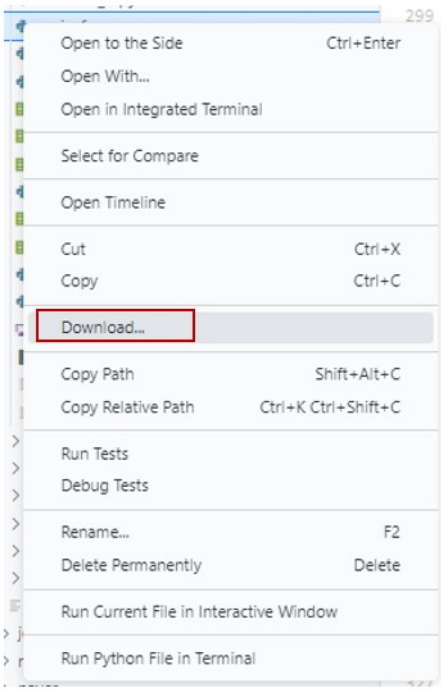
UBAI SSD 100GB . .

※ *(ubaisysadmin@uos.ac.kr)* *1TB* .

1.

VScode .

7.0.1 1.1



Download .

7.0.2 1.2

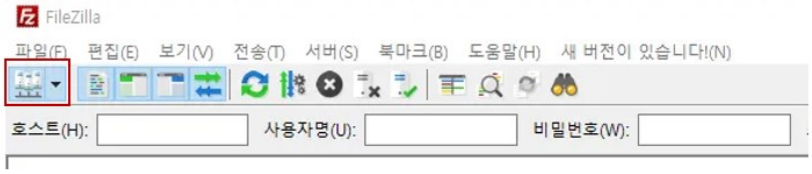
, (Explorer) .

2.

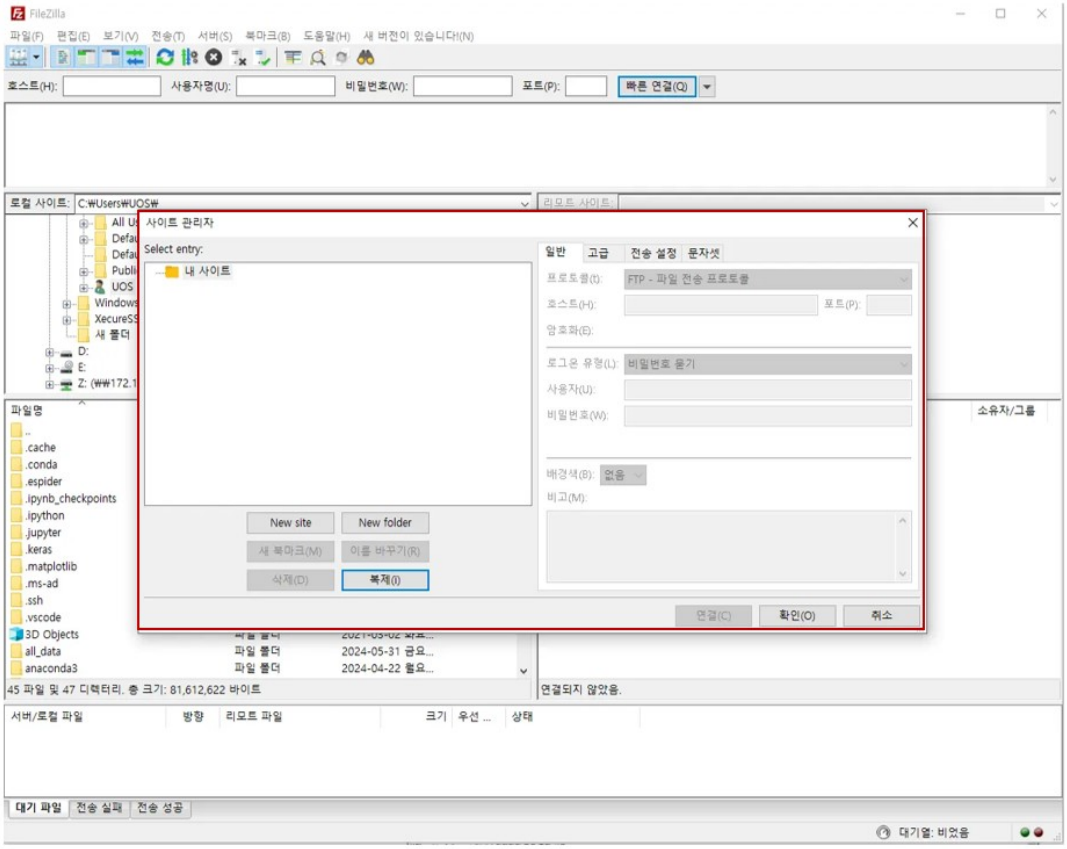
. FileZilla . FileZilla .

FileZilla , .

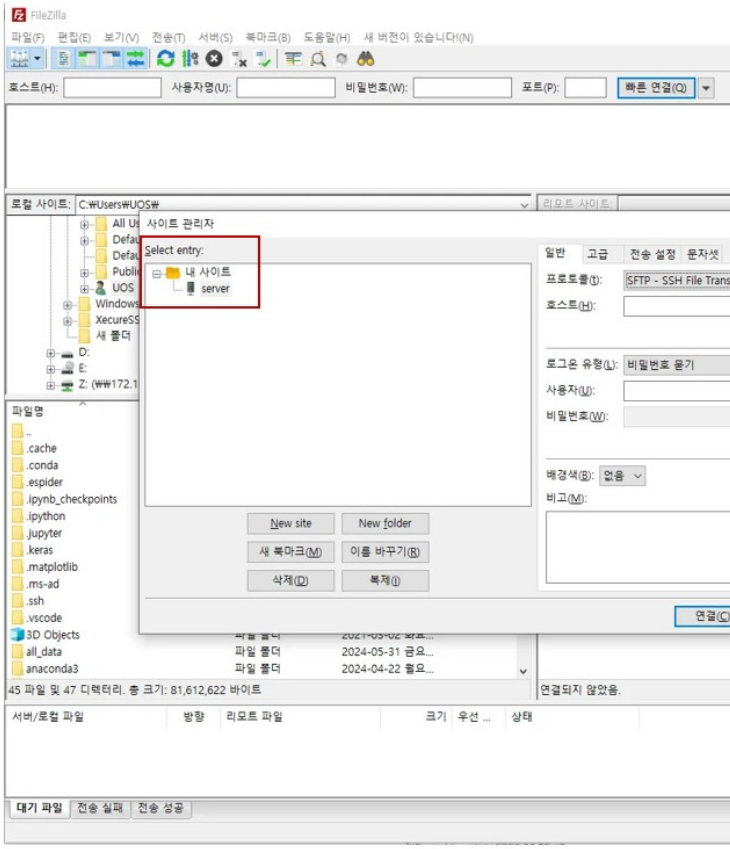
7.0.1 2.1



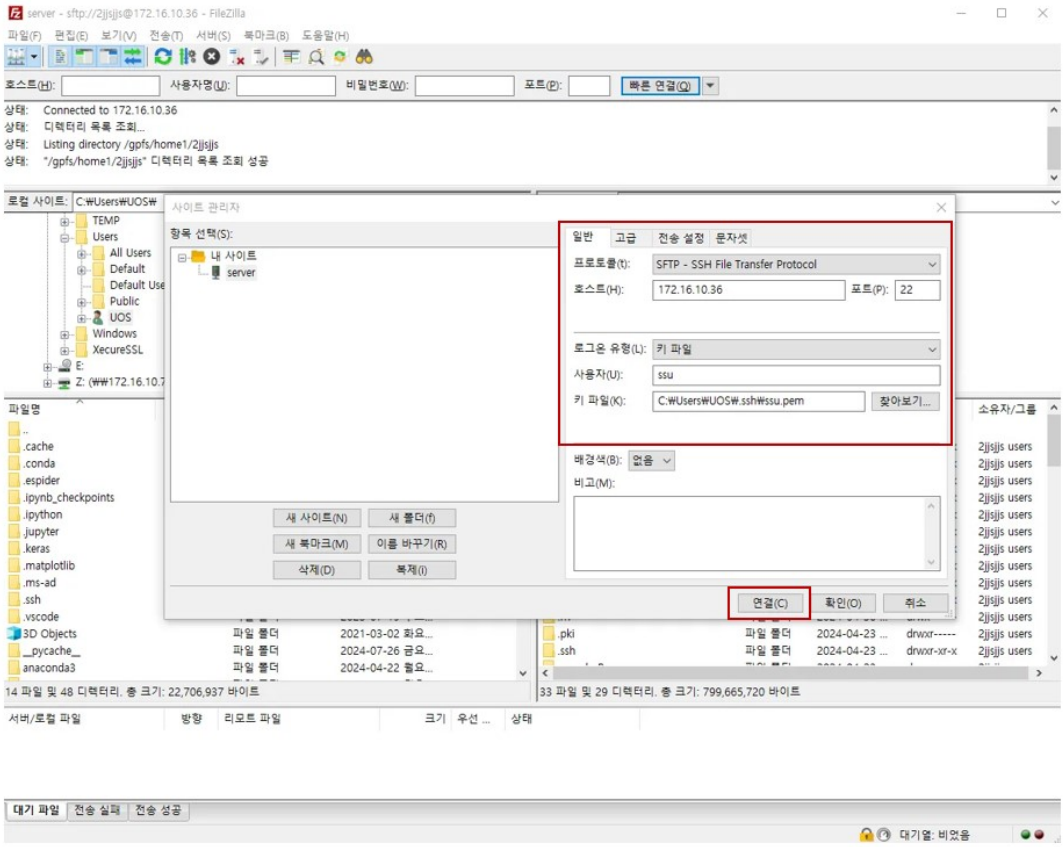
1. .


FileZilla
파일(F) 편집(E) 보기(V) 전송(T) 서버(S) 북마크(B) 도움말(H) 새 버전이 있습니다!(N)
호스트(H):
사용자명(U):
비밀번호(W):
포트(P):
빠른 연결(Q)
로컬 사이트: C:₩Users₩UOS₩
리모트 사이트:
사이트 관리자
Select entry:
내 사이트
New site
New folder
새 북마크(M)
이름 바꾸기(R)
삭제(D)
복제(I)
일반
고급
전송 설정
문자셋
프로토콜(t):
FTP - 파일 전송 프로토콜
호스트(H):
포트(P):
암호화(E):
로그온 유형(L):
비밀번호 묻기
사용자(U):
비밀번호(W):
배경색(B):
없음
비고(M):
연결(C)
확인(O)
취소
파일명
.cache
.conda
.espider
.ipynb_checkpoints
.ipython
.jupyter
.keras
.matplotlib
.ms-ad
.ssh
.vscode
3D Objects
all_data
anaconda3
파일 폴더
2024-05-31 금요...
2024-04-22 월요...
45 파일 및 47 디렉터리. 총 크기: 81,612,622 바이트
연결되지 않았음.
서버/로컬 파일
방향
리모트 파일
크기
우선 ...
상태
대기 파일
전송 실패
전송 성공
대기열: 비었음

2. New site .

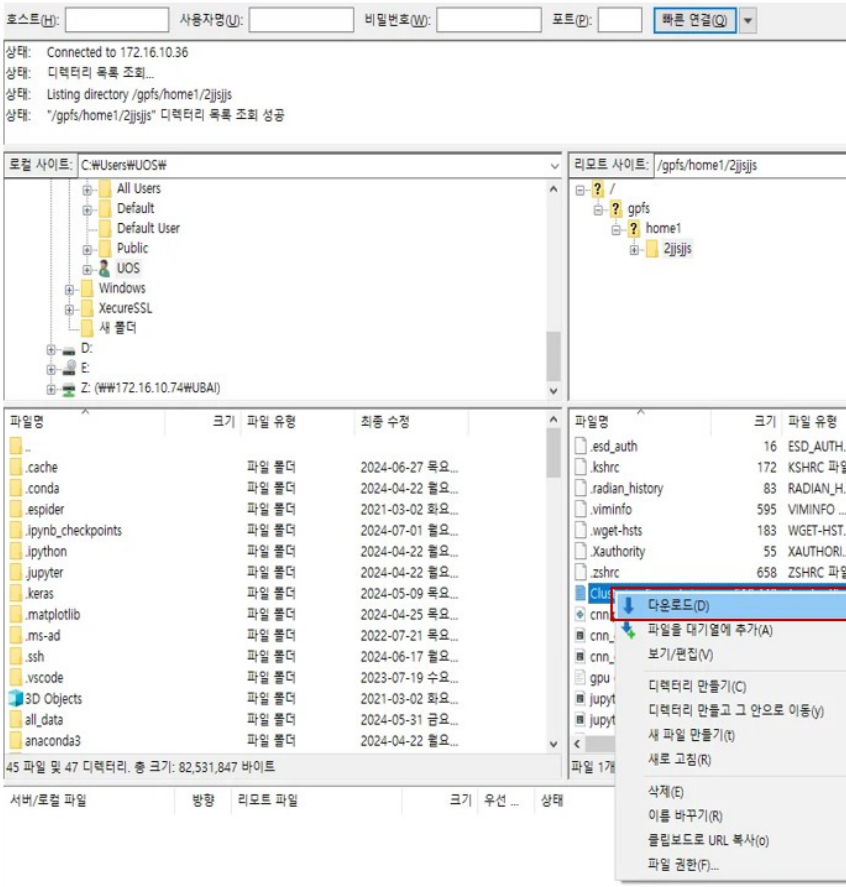


3. “SFTP - SSH File Transfer Protocol” .



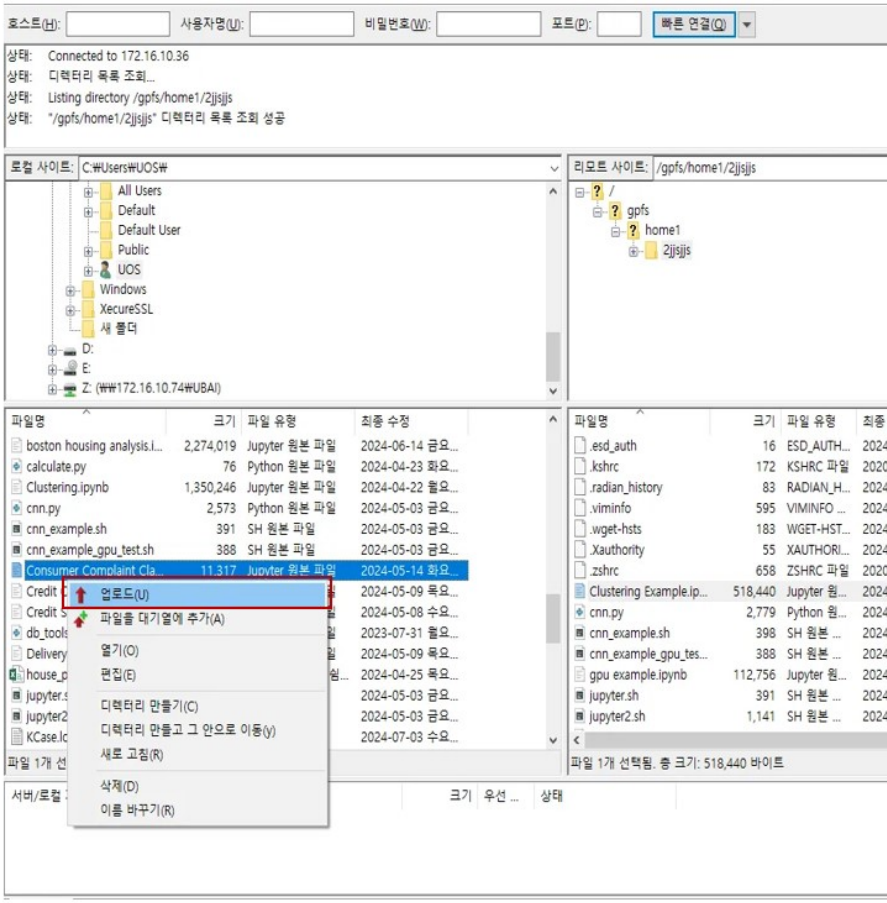
4. .

- : SFTP
- : IPv4
- : 22
- :
- : (ex. ubuntu)
- : (.pem)
-


호스트(H):
사용자명(U):
비밀번호(W):
포트(P):
빠른 연결(Q)
상태: Connected to 172.16.10.36
상태: 디렉터리 목록 조회...
상태: Listing directory /gpfs/home1/2jjsjjs
상태: "/gpfs/home1/2jjsjjs" 디렉터리 목록 조회 성공
로컬 사이트: C:₩Users₩UOS₩
All Users
Default
Default User
Public
UOS
Windows
XecureSSL
새 폴더
D:
E:
Z: (₩₩172.16.10.74₩UBAI)
리모트 사이트: /gpfs/home1/2jjsjjs
/
gpfs
home1
2jjsjjs
파일명
크기
파일 유형
최종 수정
.cache
.conda
.espider
.ipynb_checkpoints
.ipython
.jupyter
.keras
.matplotlib
.ms-ad
.ssh
.vscode
3D Objects
all_data
anaconda3
45 파일 및 47 디렉터리. 총 크기: 82,531,847 바이트
.esd_auth
.kshrc
.radian_history
.viminfo
.wget-hsts
.Xauthority
.zshrc
다운로드(D)
파일을 대기열에 추가(A)
보기/편집(V)
디렉터리 만들기(C)
디렉터리 만들고 그 안으로 이동(y)
새 파일 만들기(t)
새로 고침(R)
삭제(E)
이름 바꾸기(R)
클립보드로 URL 복사(o)
파일 권한(F)...
서버/로컬 파일
방향
리모트 파일
크기
우선 ...
상태

( ) .

7.0.2 2.2


호스트(H):
사용자명(U):
비밀번호(W):
포트(P):
빠른 연결(Q)
상태: Connected to 172.16.10.36
상태: 디렉터리 목록 조회...
상태: Listing directory /gpfs/home1/2jjsjjs
상태: "/gpfs/home1/2jjsjjs" 디렉터리 목록 조회 성공
로컬 사이트: C:₩Users₩UOS₩
All Users
Default
Default User
Public
UOS
Windows
XecureSSL
새 폴더
D:
E:
Z: (₩₩172.16.10.74₩UBAI)
리모트 사이트: /gpfs/home1/2jjsjjs
/
gpfs
home1
2jjsjjs
파일명 크기 파일 유형 최종 수정
boston housing analysis.i... 2,274,019 Jupyter 원본 파일 2024-06-14 금요...
calculate.py 76 Python 원본 파일 2024-04-23 화요...
Clustering.ipynb 1,350,246 Jupyter 원본 파일 2024-04-22 월요...
cnn.py 2,573 Python 원본 파일 2024-05-03 금요...
cnn_example.sh 391 SH 원본 파일 2024-05-03 금요...
cnn_example_gpu_test.sh 388 SH 원본 파일 2024-05-03 금요...
Consumer Complaint Cla... 11,317 Jupyter 원본 파일 2024-05-14 화요...
업로드(U)
파일을 대기열에 추가(A)
열기(O)
편집(E)
디렉터리 만들기(C)
디렉터리 만들고 그 안으로 이동(y)
새로 고침(R)
삭제(D)
이름 바꾸기(R)
파일 1개 선택됨. 총 크기: 518,440 바이트
서버/로컬
크기 우선 ... 상태

, ( ) .

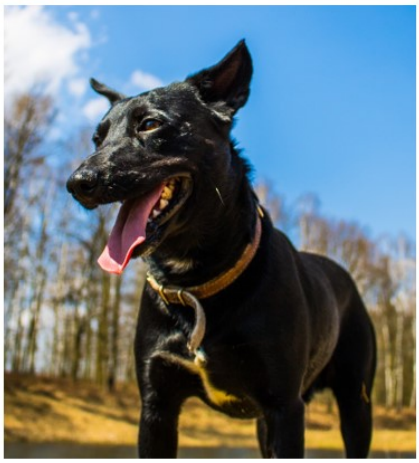
8 Chapter6.

Python .

(Image Captioning) .

, .

, , “A black dog sitting among leaves in a forest, surrounded by trees.(.)” .



A black dog sitting among leaves in a forest, surrounded by trees.

.

1.

, .

. captioning . Terminal , cd (captioning) .

`pwd` .

`cd` .

```
(captioning) [ssu@gate1 ~]$ pwd
/home1/ssu
(captioning) [ssu@gate1 ~]$ cd captioning
(captioning) [ssu@gate1 captioning]$ pwd
/home1/ssu/captioning
```

.

`conda create -n captioning python=3.8` python 3.8 captioning .

`conda activate captioning` .

```
(base) [ssu@gate1 captioning]$ conda activate captioning
(captioning) [ssu@gate1 captioning]$
```

, .

```
pip install torch torchvision transformers matplotlib
```

2.

. **Microsoft COCO (MS COCO)** .

MS COCO Object detection( ), Segmentation( ), Captioning ,
.

MS COCO shell .

```
#!/bin/bash

# COCO dataset directory
mkdir -p /data/coco

# Download COCO Train2014 images and captions
cd /data/coco
wget http://images.cocodataset.org/zips/train2014.zip
wget http://images.cocodataset.org/zips/val2014.zip
wget http://images.cocodataset.org/annotations/annotations_trainval2014.zip

# Unzip the dataset
unzip train2014.zip
unzip val2014.zip
unzip annotations_trainval2014.zip
```

`mkdir` data . mkdir .

`cd` .

`wget` MS COCO dataset , .

`unzip` dataset zip , .

, shell .

```
(base) [ssu@gate1 captioning]$ ./dataset_download.sh
bash: ./dataset_download.sh: Permission denied
(base) [ssu@gate1 captioning]$ chmod +x dataset_download.sh
```

`datset_download.sh` shell , “permission denied ( )” . , `chmod` .

`chmod` , .

```
chmod [references][operator][modes] file1 ...
```

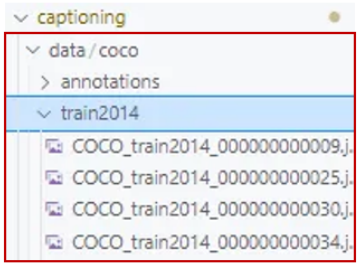
r | (read) |
w | (write) |
x | (execute)|

`chmod +x [file_name.sh]` +x [file_name.sh] .

```
(base) [ssu@gate1 captioning]$ ./dataset_download.sh
bash: ./dataset_download.sh: Permission denied
(base) [ssu@gate1 captioning]$ chmod +x dataset_download.sh
```

.

, .



3.

, .

Transformer . Transformer 2017 Google , .

transformer.py .

```python
import os
import json
import torch
import torch.nn as nn
import torch.optim as optim
import torchvision.transforms as transforms
from torch.utils.data import DataLoader, Dataset
from PIL import Image
from transformers import ViTModel, BertTokenizer, BertConfig, BertModel

from tqdm import tqdm

#      (GPU  )
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

#
data_dir = 'data/coco/train2014'  #
ann_file = 'data/coco/annotations/captions_train2014.json'  #

#
class CocoDataset(Dataset):
    def __init__(self, data_dir, ann_file, transform=None):
        self.data_dir = data_dir
        self.transform = transform
        with open(ann_file, 'r') as f:
            self.annotations = json.load(f)['annotations']
        self.image_ids = [item['image_id'] for item in self.annotations]
        self.captions = [item['caption'] for item in self.annotations]
    def __len__(self):
        return len(self.annotations)
    def __getitem__(self, idx):
        image_id = self.image_ids[idx]
        img_path = os.path.join(self.data_dir, f'COCO_train2014_{image_id:012}.jpg')
        image = Image.open(img_path).convert("RGB")
        caption = self.captions[idx]
        if self.transform:
            image = self.transform(image)
        return image, caption

#
transform = transforms.Compose([
    transforms.Resize((224, 224)),
```

```
    transforms.ToTensor(),
])

#
dataset = CocoDataset(data_dir=data_dir, ann_file=ann_file, transform=transform)
data_loader = DataLoader(dataset, batch_size=32, shuffle=True)

# Transformer
class TransformerImageCaptioning(nn.Module):
    def __init__(self, vocab_size):
        super(TransformerImageCaptioning, self).__init__()
        # Vision Transformer for image feature extraction
        self.vit = ViTModel.from_pretrained("google/vit-base-patch16-224-in21k")
        # Transformer Decoder for caption generation
        self.tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")
        config = BertConfig(
            vocab_size=vocab_size,
            num_hidden_layers=6,
            num_attention_heads=12,
            hidden_size=768,
            is_decoder=True,
            add_cross_attention=True
        )
        self.decoder = BertModel(config)
        # Fully connected layer to convert image features to the decoder input size
        self.fc = nn.Linear(self.vit.config.hidden_size, config.hidden_size)
        self.fc_out = nn.Linear(config.hidden_size, vocab_size)
    def forward(self, image, caption_ids):
        # Get image features from ViT
        image_features = self.vit(image).last_hidden_state
        # Average pooling to keep the batch dimension
        image_features = torch.mean(image_features, dim=1)
        image_features = self.fc(image_features)
        # Repeat the image features across the sequence length
        image_features = image_features.unsqueeze(1).repeat(1, caption_ids.size(1), 1)
        # Get text features from decoder (BERT model in decoder mode)
        attention_mask = (caption_ids != self.tokenizer.pad_token_id).float()
        decoder_outputs = self.decoder(input_ids=caption_ids, attention_mask=attention_mask,
        logits = self.fc_out(decoder_outputs.last_hidden_state)
        return logits

#        ,
```

```
vocab_size = 30522  # BERT       vocab_size
model = TransformerImageCaptioning(vocab_size).to(device)
criterion = nn.CrossEntropyLoss(ignore_index=model.tokenizer.pad_token_id)
optimizer = optim.Adam(model.parameters(), lr=5e-5)


#
def train_model(data_loader, model, criterion, optimizer, num_epochs=5):
    model.train()
    for epoch in range(num_epochs):
        print(f"Starting epoch {epoch + 1}/{num_epochs}")
        epoch_loss = 0
        progress_bar = tqdm(data_loader, desc=f"Epoch {epoch + 1}")
        for i, (images, captions) in enumerate(progress_bar):
            images = images.to(device)
            # Tokenize captions
            caption_ids = model.tokenizer(captions, return_tensors='pt', padding=True, trunca
            optimizer.zero_grad()
            outputs = model(images, caption_ids)
            # Align dimensions
            outputs = outputs.view(-1, vocab_size)
            caption_ids = caption_ids.view(-1)
            loss = criterion(outputs, caption_ids)
            loss.backward()
            optimizer.step()
            epoch_loss += loss.item()
            progress_bar.set_postfix(loss=loss.item())
        print(f'Epoch [{epoch + 1}/{num_epochs}], Loss: {epoch_loss / len(data_loader):.4f}'
    print("Training completed")

train_model(data_loader, model, criterion, optimizer, num_epochs=5)

#
model_save_path = './model/transformer_image_captioning_model.pth'
torch.save(model.state_dict(), model_save_path)
print(f"Model saved at {model_save_path}")
```

slurm . , HPC slurm `transformer.sh` .

```
#!/bin/bash
#SBATCH --job-name=captioning
#SBATCH --output=./output/training_captioning_%n_%j.out
```

```
#SBATCH --error=./output/training_captioning_%n_%j.err
#SBATCH --nodes=2
#SBATCH --partition=gpu3
#SBATCH --gres=gpu:4
#SBATCH --cpus-per-task=16
#SBATCH --mem=128G
#SBATCH --time=24:00:00

echo "start at:" `date` #
echo "node: $HOSTNAME" #
echo "jobid: $SLURM_JOB_ID" # jobid

# Load modules
module load cuda/11.8

# Train the transformer-based image captioning model
python transformer.py
```

#SBATCH --job-name=captioning job-name captioning .

output error output training_captioning .

#SBATCH --nodes=2 , node 2 . , node 2 .

#SBATCH --gres=gpu:4 gpu 4 .

module load cuda/11.8 module cuda 11.8 version .

python transformer.py `transformer.py` .

`sbatch transformer.sh` (job) .

tqdm , error .

```
14080   Epoch 1:  54%|[redacted]  | 7039/12942 [1:01:55<50:41,  1.94it/s, loss=0.05]
14081   Epoch 1:  54%|[redacted]  | 7039/12942 [1:01:55<50:41,  1.94it/s, loss=0.00199]
14082   Epoch 1:  54%|[redacted]  | 7040/12942 [1:01:55<51:08,  1.92it/s, loss=0.00199]
14083   Epoch 1:  54%|[redacted]  | 7040/12942 [1:01:56<51:08,  1.92it/s, loss=0.0475]
14084   Epoch 1:  54%|[redacted]  | 7041/12942 [1:01:56<51:23,  1.91it/s, loss=0.0475]
14085   Epoch 1:  54%|[redacted]  | 7041/12942 [1:01:56<51:23,  1.91it/s, loss=0.00561]
```

out log , epoch (loss) . . .

```
start at: Mon Oct 28 14:07:42 KST 2024
node: n063
jobid: 247333
Starting epoch 1/5
Epoch [1/5], Loss: 0.1303
Starting epoch 2/5
Epoch [2/5], Loss: 0.0072
Starting epoch 3/5
Epoch [3/5], Loss: 0.0025
Starting epoch 4/5
Epoch [4/5], Loss: 0.0006
Starting epoch 5/5
Epoch [5/5], Loss: 0.0001
Training completed
Model saved at ./model/transformer_image_captioning_model.pth
```

4.

. . .

(1) , . coco dataset val2014 dataset .

(2) , .

(3) .

, .

```
import os
import json
import torch
import torch.nn as nn
import torch.optim as optim
import torchvision.transforms as transforms
from torch.utils.data import DataLoader, Dataset
from PIL import Image
from transformers import ViTModel, BertTokenizer, BertConfig, BertModel
from tqdm import tqdm

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

#
```

```
def load_model(model_path, vocab_size):
    model = TransformerImageCaptioning(vocab_size).to(device)
    state_dict = torch.load(model_path, map_location=device)
    model.load_state_dict(state_dict)
    model.eval()
    return model

#
def load_validation_data(data_dir, ann_file, transform):
    class CocoDataset(Dataset):
        def __init__(self, data_dir, ann_file, transform=None):
            self.data_dir = data_dir
            self.transform = transform
            with open(ann_file, 'r') as f:
                self.annotations = json.load(f)['annotations']
            self.image_ids = [item['image_id'] for item in self.annotations]
            self.captions = [item['caption'] for item in self.annotations]
        def __len__(self):
            return len(self.annotations)
        def __getitem__(self, idx):
            image_id = self.image_ids[idx]
            img_path = os.path.join(self.data_dir, f'COCO_val2014_{image_id:012}.jpg')
            image = Image.open(img_path).convert("RGB")
            caption = self.captions[idx]
            if self.transform:
                image = self.transform(image)
            return image, caption, image_id

    dataset = CocoDataset(data_dir, ann_file, transform)
    data_loader = DataLoader(dataset, batch_size=1, shuffle=False)
    return data_loader

#
def validate_model(model, data_loader):
    model.eval()
    tokenizer = model.tokenizer
    total_loss = 0
    criterion = nn.CrossEntropyLoss(ignore_index=tokenizer.pad_token_id)
    with torch.no_grad():
        for images, captions, image_ids in tqdm(data_loader, desc="Validating"):
            images = images.to(device)
            caption_ids = tokenizer(captions, return_tensors='pt', padding=True, truncation=
```

```
            outputs = model(images, caption_ids)
            outputs = outputs.view(-1, tokenizer.vocab_size)
            caption_ids = caption_ids.view(-1)
            loss = criterion(outputs, caption_ids)
            total_loss += loss.item()
    avg_loss = total_loss / len(data_loader)
    print(f'Validation Loss: {avg_loss:.4f}')

#
data_dir = 'data/coco/val2014'
ann_file = 'data/coco/annotations/captions_val2014.json'
transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
])
validation_loader = load_validation_data(data_dir, ann_file, transform)

#
vocab_size = 30522
model_path = './model/transformer_image_captioning_model.pth'
model = load_model(model_path, vocab_size)
validate_model(model, validation_loader)
```

shell `sbatch val.py` (job) . out , (Loss) 0.0073 .

```
start at: Tue Oct 29 11:13:22 KST 2024
node: n083
jobid: 247856
Validation Loss: 0.0073
```