PHISON

Phison Electronics Corporation aiDAPTIVLink2.0 NXUN203.00 Install SOP & User manual

Version 1.1

Phison Electronics Corporation
Tel: +886-37-586-896 Fax: +886-37-587-868
E-mail: sales@phison.com / suppport@phison.com

REVISION HISTORY

Revision	Draft Date	History	aiDAPTIVLink Version	Author
1.0	2025/06/27	First Release	NXUN203.00	Heine Chu
1.1	2025/07/18	Revised Content	NXUN203.00	Heine Chu

TABLE OF CONTENTS

REVISION HISTORY **3**

TABLE OF CONTENTS **4**

LIST OF FIGURES **7**

LIST OF TABLES **9**

1. CHECK DEVICE **11**

1.1. Check GPU is in Approved Vendor List? (AVL) 11

1.2. Check the number of GPUs 11

1.3. Check DRAM and aiDAPTIVCache from different LLM model size 11

1.4. Check CPU configuration 12

2. INSTALLATION AND PREPARATION **13**

2.1. Environment Preparation 13

2.1.1. Requirements 13

2.1.2. Install GPU Driver 13

2.1.3. Install GPU Toolkit 14

2.1.4. Install GPU Toolkit 15

2.2. aiDAPTIVLink Installation 15

2.2.1. Deploy aiDAPTIV 15

2.2.2. Disk Setup (LVM Setting) 16

2.2.3. FW update 18

2.3. Login Huggingface 20

2.3.1. How to get Hugging Face token 20

2.3.2. How to register the Hugging Face token 23

2.4. Download Llama-3.1-8B-Instruct 23

2.4.1. Download Llama-3.1-8B-Instruct model 24

2.5. Run aiDAPTIV 25

2.5.1. Start training your model 25

2.5.2. text-generation settings 26

2.5.3. Monitor training status 27

2.5.4. Successful example 28

2.5.5. image-text-to-text settings 29

2.5.6. "phisonai2" command arguments 31

2.6. Quick Start for pytorch 36

2.6.1. Without aiDAPTIVLink 36

2.6.2. With aiDAPTIVLink 36

3. PERFORMANCE RESULT 37

3.1. aiDAPTIV Toolkit Installation flow 37

3.1.1. Download aiDAPTIV Toolkit 37

3.1.2. Unzip download file 37

3.1.3. Change to folder 37

3.1.4. Deploy aiDAPTIV Toolkit and related library 37

3.2. Start aiDAPTIV Toolkit 38

3.2.1. Modify training setting in aiDAPTIV_Toolkit_2.3.0/project.ini 38

3.2.2. Enter aiDAPTIV Toolkit folder 38

3.2.3. Run aiDAPTIV Toolkit Performance Test 39

3.3. Performance Result 40

3.3.1. Log directory Description 40

3.3.2. Performance Reference 41

4. DATASET CONFIGURATION 42

4.1. QA datasets 42

4.2. Pretrain datasets 43

4.3. RAG datasets 44

4.4. Description for each value 45

4.4.1. Example 1 (QA dataset) 46

4.4.2. Example 2 (RAG dataset) 46

4.4.3. Example 3 (nested RAG dataset) 47

4.5. Compare Change 48

4.5.1. Previous dataloader on Dahoas/rm-static after apply template 48

4.5.2. Current dataloader on Dahoas/rm-static after apply template 48

5. MULTIMODALITY DATASET CONFIGURATION 49

5.1. VQA datasets 49

5.2. Description for each value 49

5.3. Supported dataset 50

*5.3.1. Custom dataset: The data path should be in the format of ./.json or ./.csv** to properly parse the data 50*

5.3.2. Example (MathVista QA dataset) 51

APPENDIX A. HOW TO EVALUATE TRAINED MODEL 52

A.1 Quick Start 52

A.2 Prepare Your Evaluation Data 53

A.3 Prepare Your Evaluation Function (Advance) 53

APPENDIX B. HOW TO TRAIN MODEL IN DOCKER 57

B.1 Docker Installation option 57

B.2 Run aiDAPTIV Image 58

APPENDIX C. HOW TO SET SWAP FILE 59

APPENDIX D. APPROVED VENDOR LIST (AVL) 60

D.1 GPU AVL 60

D.2 CPU AVL 60

D.3 Support Model list 61

LIST OF FIGURES

Figure 2-1 GPU Driver Installation 14
Figure 2-2 Deploy aiDAPTIV+ 15
Figure 2-3 Firmware update 15
Figure 2-4 aiDAPTIVLink file 16
Figure 2-5 Deploy aiDAPTIV+ successful 16
Figure 2-6 aiDAPTIV2 folder 16
Figure 2-7 Check aiDAPTIVLink version 16
Figure 2-8 Clear disk 16
Figure 2-9 Create LVM 17
Figure 2-10 LVM 17
Figure 2-11 FW update 18
Figure 2-12 FW check/update 18
Figure 2-13 Select check/update FW 18
Figure 2-14 Add nvme_path 18
Figure 2-15 Repeat while more than one disk 19
Figure 2-16 FW update check 19
Figure 2-17 FW updated 19
Figure 2-18 Login huggingface 20
Figure 2-19 selec setting 20
Figure 2-20 Access tokens 21
Figure 2-21 selec setting 21
Figure 2-22 Get tokens 22
Figure 2-23 Get tokens in account 22
Figure 2-24 Login huggingface 23
Figure 2-25 Download Llama-3.1-8B-Instruct 23
Figure 2-26 Model Name 24
Figure 2-27 Download Llama-3.1-8B-Instruct successful 24
Figure 2-28 Weight of the model 24
Figure 2-29 Location of model 25
Figure 2-30 Location of SSD mount path 25

Figure 2-31 aiDAPTIV training log 28
Figure 2-32 Finetuned model 28
Figure 2-33 Import optimizer 36
Figure 2-34 Import optimizer 36
Figure 2-35 Import library 36
Figure 2-36 Create model instance and call initialize 36
Figure 3-1 aiDAPTIV Toolkit and related library 37
Figure 3-2 aiDAPTIV Toolkit Performance Test 39
Figure 3-3 Ram Used 40
Figure 3-4 Training Log 40
Figure 3-5 Example of Performance Result 41
Figure 3-6 Llama-3.1-8B-Instruct’s Performance Result on 4000ada*4 41
Figure 3-7 Llama-3.1-8B-Instruct’s Performance Result on A6000*8 41
Figure A-1 Log file 56
Figure B-1 docker image 57
Figure B-2 docker image list 57
Figure B-3 docker successful example 58

LIST OF TABLES

Table 1-1 Recommend Configuration 11
Table 2-1 Recommend environment 13
Table E-1 GPU AVL 60
Table E-2 CPU AVL 60
Table E-3 Support model list 61

Preface

Purpose

This manual is intended solely for aiDAPTIV partners.

- aiDAPTIVLink Version: NXUN203.00
- aiDAPTIVCache Family：

Model
AI100E SSD

1. CHECK DEVICE

In this section, we will be checking system specifications (GPU/CPU/RAM/aiDAPTIVCache) before starting to use aiDAPTIVLink. This section will detail which GPU/CPU are in Phison AVL, and how much RAM is required for the different LLM models.

1.1. Check GPU is in Approved Vendor List? (AVL)

- Check GPU is in AVL list (Appendix D - GPU AVL).
- Needs to do validation if GPU is not in AVL.
- It is recommended that the GPU must be connected to the PCIe Slot of X16. Connecting other Slots (ex: X8, X4) will result in GPU performance degradation.

1.2. Check the number of GPUs

- Number of GPUs = 2^n (n=0,1,2,3,4, GPUs = 1,2,4,8,16)

1.3. Check DRAM and aiDAPTIVCache from different LLM model size

- DRAM and aiDAPTIVCache requirements from different LLM model sizes

Table 1-1 Recommend Configuration

LLM model size	**≦13B**	**<34B**	**<70B**	**<180B**
DRAM	64GB	64GB	128GB	128GB
aiDAPTIVCache capacity	1TB	1TB	2TB	2TB
AiDAPTIVCache count	1	2	2	4
aiDAPTIVCache slot count	1	2	2	4

  - Rack server: Use U.2 slot.
  - Recommend Gen4 or above.
  - Recommend DRAM 2933MHz or above.
  - Recommended DRAM size is 128GB or more
  - Recommended DRAM channel number is 8 or more, ex: 16GB x8

1.4. Check CPU configuration

- Check CPU against AVL list (Appendix D - CPU AVL).
- Recommend CPU Cores: 8 or above

CPU lanes calculate by GPU count and aiDAPTIVCache count.
Total PCIe lanes = GPU count * 16 + aiDAPTIVCache count * 4

*Example : For GPU count =4 and 70B LLM model size (aiDAPTIVCache 2)

Total 4 * 16 + 2 * 4 = 72 lanes, It means at least 72 lanes for optimal performance.

2. INSTALLATION AND PREPARATION

In this section, we will start installing the aiDAPTIVLink and explain how to use it for Domain training or fine-tuning.

2.1. Environment Preparation

2.1.1. Requirements

- Recommend environment

Table 2-1 Recommend environment

Category	**Detail**
OS	Ubuntu 22.04.3 Desktop
GPU driver	Nvidia driver: version 550 installation
Python	3.10.12

2.1.2. Install GPU Driver

- Install Driver (Estimated time: 5 min)
  - Install NVIDIA Driver

```
sudo apt install nvidia-utils-550

sudo apt install nvidia-driver-550
```

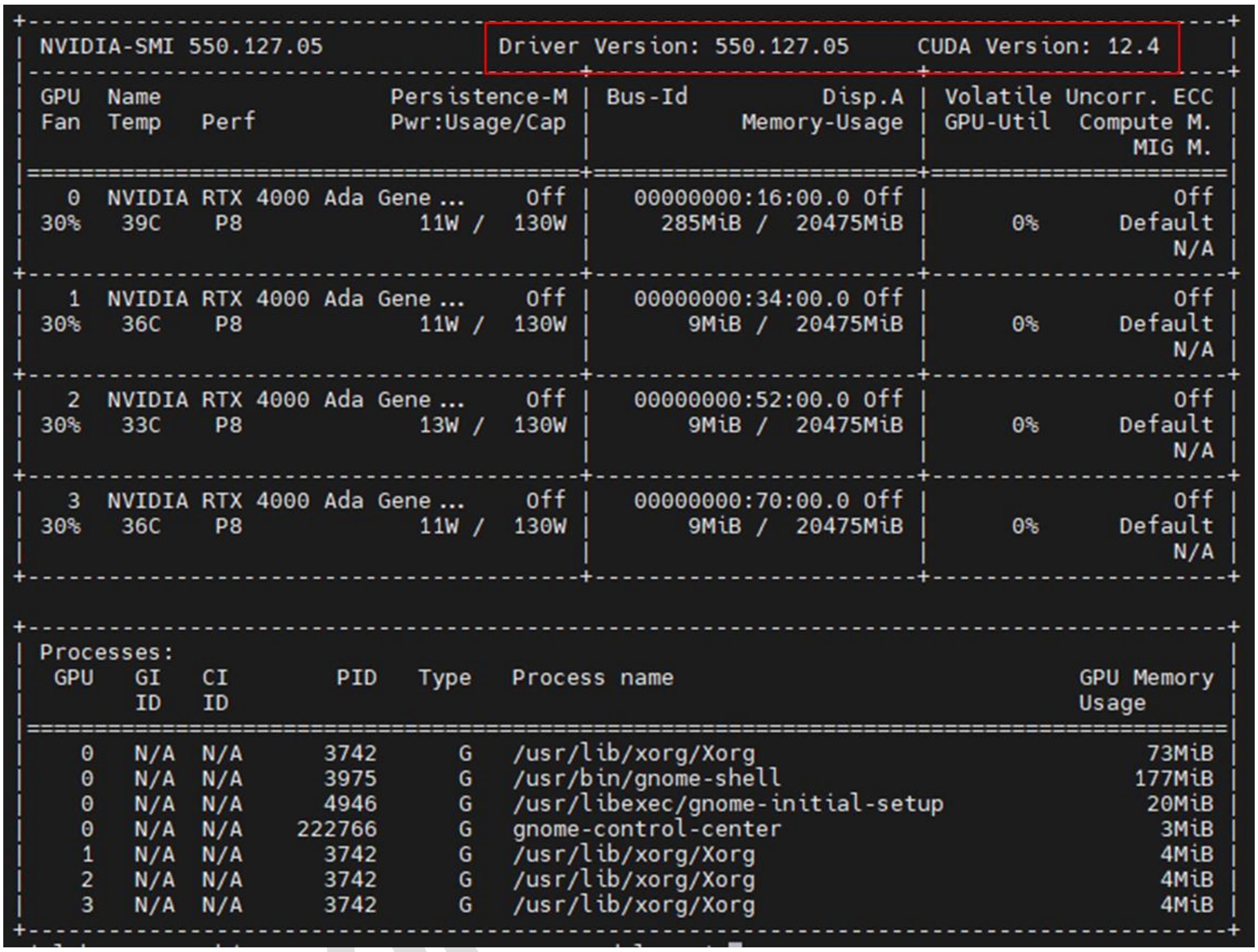
- Reboot system

```
sudo reboot
```

- Successful example

```
nvidia-smi
```

Verify GPU driver and CUDA version in the resulting log.



```
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 550.127.05             Driver Version: 550.127.05     CUDA Version: 12.4     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  NVIDIA RTX 4000 Ada Gene ...    Off |   00000000:16:00.0 Off |                  Off |
| 30%   39C    P8             11W /  130W |     285MiB /  20475MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+
|   1  NVIDIA RTX 4000 Ada Gene ...    Off |   00000000:34:00.0 Off |                  Off |
| 30%   36C    P8             11W /  130W |       9MiB /  20475MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+
|   2  NVIDIA RTX 4000 Ada Gene ...    Off |   00000000:52:00.0 Off |                  Off |
| 30%   33C    P8             13W /  130W |       9MiB /  20475MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+
|   3  NVIDIA RTX 4000 Ada Gene ...    Off |   00000000:70:00.0 Off |                  Off |
| 30%   36C    P8             11W /  130W |       9MiB /  20475MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+

+-----------------------------------------------------------------------------------------+
| Processes:                                                                              |
|  GPU   GI   CI        PID   Type   Process name                              GPU Memory |
|        ID   ID                                                               Usage      |
|=========================================================================================|
|    0   N/A  N/A      3742      G   /usr/lib/xorg/Xorg                             73MiB |
|    0   N/A  N/A      3975      G   /usr/bin/gnome-shell                          177MiB |
|    0   N/A  N/A      4946      G   /usr/libexec/gnome-initial-setup               20MiB |
|    0   N/A  N/A    222766      G   gnome-control-center                            3MiB |
|    1   N/A  N/A      3742      G   /usr/lib/xorg/Xorg                              4MiB |
|    2   N/A  N/A      3742      G   /usr/lib/xorg/Xorg                              4MiB |
|    3   N/A  N/A      3742      G   /usr/lib/xorg/Xorg                              4MiB |
+-----------------------------------------------------------------------------------------+
```

Figure 2-1 GPU Driver Installation

2.1.3. *Install GPU Toolkit*

- Install NVIDIA Toolkit: CUDA

```
wget
https://developer.download.nvidia.com/compute/cuda/12.4.1/local_installers/cuda_12.4.1_550.54.15_linux.run

sudo sh cuda_12.4.1_550.54.15_linux.run

# Continue > accept > Look at the images below and do not check the box

# Turn [X] Driver --> [ ] Driver

# Then, move the cursor to Install and press Enter.
```

2.1.4. *Install GPU Toolkit*

- Install NVIDIA Library: cuDNN

```
wget https://developer.download.nvidia.com/compute/cudnn/9.4.0/local_installers/cudnn-local-repo-ubuntu2204-9.4.0_1.0-1_amd64.deb
sudo dpkg -i cudnn-local-repo-ubuntu2204-9.4.0_1.0-1_amd64.deb
sudo cp /var/cudnn-local-repo-ubuntu2204-9.4.0/cudnn-*-keyring.gpg
/usr/share/keyrings/
sudo apt-get update
sudo apt-get -y install cudnn-cuda-12
```

2.2. aiDAPTIVLink Installation

2.2.1. *Deploy aiDAPTIV*

To avoid system conflicts please install aiDAPTIVLink on a fresh Ubuntu system. For existing systems or to avoid potential conflicts a Docker version can be installed (Chapt D).

- Native Installation option (The deployed user can be a custom user or root.)
    - Setup tool (aiDAPTIVLink)

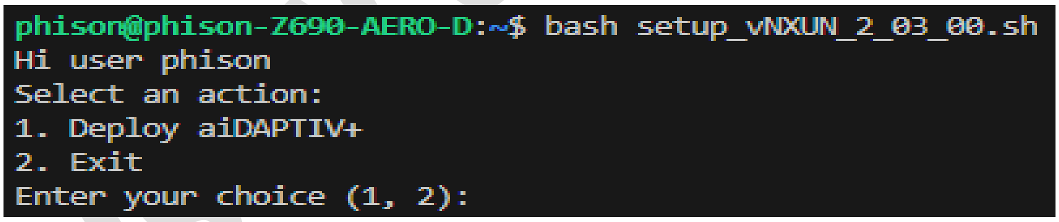
```
wget https://phisonbucket.s3.ap-northeast-1.amazonaws.com/setup_vNXUN_2_03_00.sh
```

    - Deploy aiDAPTIV (Install aiDAPTIVLink)

```
bash setup_vNXUN_2_03_00.sh
```

- Select "1. Deploy aiDAPTIV+".

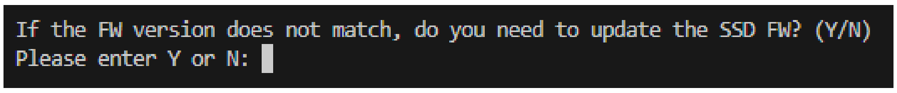
  Note: The options may differ in this stage depending on whether you've installed before.



```
phison@phison-Z690-AERO-D:~$ bash setup_vNXUN_2_03_00.sh
Hi user phison
Select an action:
1. Deploy aiDAPTIV+
2. Exit
Enter your choice (1, 2):
```

Figure 2-2 Deploy aiDAPTIV+

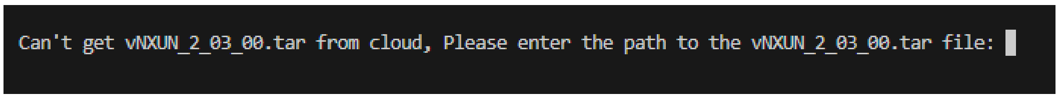
- If you can see the option "FW Update" in the former step.
  You can select it and see the picture below..
  Select 'Y', the firmware update process will be initiated. And please refer to "2.2.3 FW Update" to continue the update steps. After finish FW update, please go back here and continue the install process.
  Selecting 'N' will directly proceed with the installation.
  However, if you don't have the "FW update" option in the former step and want to update the FW, please wait until you finish the installation and check again if the option "2.2.3 FW update" has shown up or not.



```
If the FW version does not match, do you need to update the SSD FW? (Y/N)
Please enter Y or N:
```

Figure 2-3 Firmware update

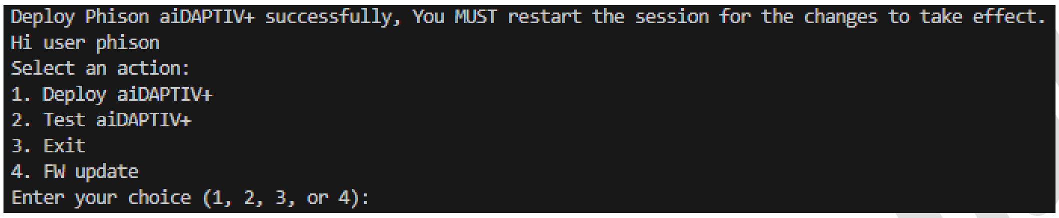
- If you can't get vNXUN_2_03_00.tar from cloud, please enter the path to the vNXUN_2_03_00.tar file



```
Can't get vNXUN_2_03_00.tar from cloud, Please enter the path to the vNXUN_2_03_00.tar file:
```

Figure 2-4 aiDAPTIVLink file

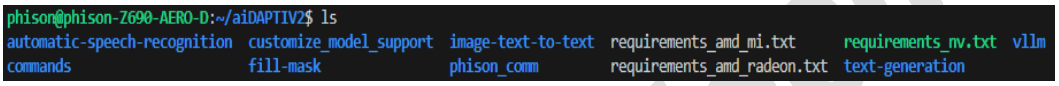
- Successful example
  - It will show the following message.



```
Deploy Phison aiDAPTIV+ successfully, You MUST restart the session for the changes to take effect.
Hi user phison
Select an action:
1. Deploy aiDAPTIV+
2. Test aiDAPTIV+
3. Exit
4. FW update
Enter your choice (1, 2, 3, or 4):
```

Figure 2-5 Deploy aiDAPTIV+ successful

  - When complete there would be a "aiDAPTIV2" folder /home/$user/

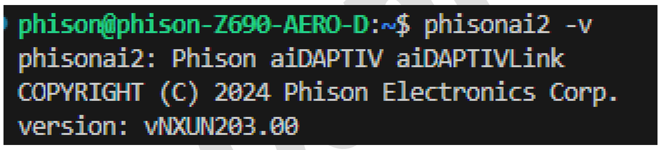


```
phison@phison-Z690-AERO-D:~/aiDAPTIV2$ ls
automatic-speech-recognition  customize_model_support  image-text-to-text  requirements_and_mi.txt     requirements_nv.txt  vllm
commands                      fill-mask                phison_comm         requirements_and_radeon.txt  text-generation
```

Figure 2-6 aiDAPTIV2 folder

- Check aiDAPTIVLink verison:

```
phisonai2 -v
```



```
phison@phison-Z690-AERO-D:~$ phisonai2 -v
phisonai2: Phison aiDAPTIV aiDAPTIVLink
COPYRIGHT (C) 2024 Phison Electronics Corp.
version: vNXUN203.00
```

Figure 2-7 Check aiDAPTIVLink version

2.2.2. *Disk Setup (LVM Setting)*

- Install LVM

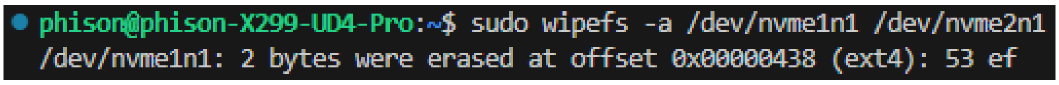
```
sudo apt update;sudo apt install lvm2 xfsprogs
```

- Check disks locations:

```
lshw -class disk -class storage | grep -E 'ai100|logical name|version: EIFZ'
lsblk | grep nvme
# Confirm ai100 device names are, for example, nvme6n1 and nvme8n1. If not, make the n
ecessary changes below to adapt as appropriate.
```

- Clear disks just in case

```
sudo wipefs -a /dev/nvme1n1 /dev/nvme2n1
```

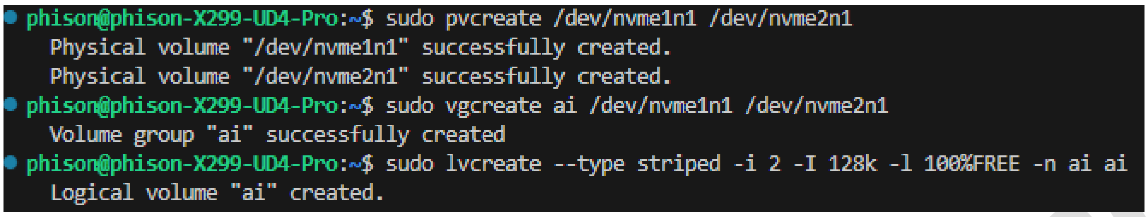


```
phison@phison-X299-UD4-Pro:~$ sudo wipefs -a /dev/nvme1n1 /dev/nvme2n1
/dev/nvme1n1: 2 bytes were erased at offset 0x00000438 (ext4): 53 ef
```

Figure 2-8 Clear disk

- Create LVM

```
sudo pvcreate /dev/nvme1n1 /dev/nvme2n1
sudo vgcreate ai /dev/nvme1n1 /dev/nvme2n1
sudo lvcreate --type striped -i 2 -I 128k -l 100%FREE -n ai ai
```



```
phison@phison-X299-UD4-Pro:~$ sudo pvcreate /dev/nvme1n1 /dev/nvme2n1
  Physical volume "/dev/nvme1n1" successfully created.
  Physical volume "/dev/nvme2n1" successfully created.
phison@phison-X299-UD4-Pro:~$ sudo vgcreate ai /dev/nvme1n1 /dev/nvme2n1
  Volume group "ai" successfully created
phison@phison-X299-UD4-Pro:~$ sudo lvcreate --type striped -i 2 -I 128k -l 100%FREE -n ai ai
  Logical volume "ai" created.
```

Figure 2-9 Create LVM

- Mount LVM

```
#Format the disk.
sudo mkfs.xfs -f -s size=4k -m crc=0 /dev/ai/ai -f
#Mount the disk.
sudo mkdir -p /mnt/nvme0
sudo mount /dev/ai/ai /mnt/nvme0
sudo chown -R $USER:$USER /mnt/nvme0
```

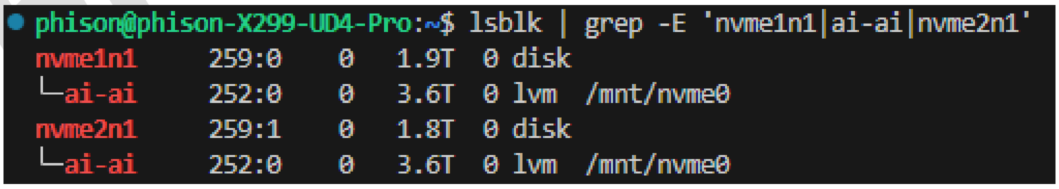
- (optional) Make mount persistent

```
# Make mount persistent
sudo echo '/dev/ai/ai /mnt/nvme0 xfs defaults,nofail 0 0' | sudo tee -a /etc/fstab

# Remove permanent mount setting
sudo sed -i '/\/dev\/ai\/ai/d' /etc/fstab
```

- Successful example

```
lsblk
```

If LVM setting is successful, you will see the following successful configuration when input “lsblk”.



```
phison@phison-X299-UD4-Pro:~$ lsblk | grep -E 'nvme1n1|ai-ai|nvme2n1'
nvme1n1      259:0    0   1.9T  0 disk
└─ai-ai      252:0    0   3.6T  0 lvm  /mnt/nvme0
nvme2n1      259:1    0   1.8T  0 disk
└─ai-ai      252:0    0   3.6T  0 lvm  /mnt/nvme0
```

Figure 2-10 LVM

- (optional) If you need to dissolve LVM Setting

```
sudo umount /mnt/nvme0;sudo lvremove -y ai;sudo pvremove -y /dev/nvme1n1 /dev/nvme2n1 --
force --force
```

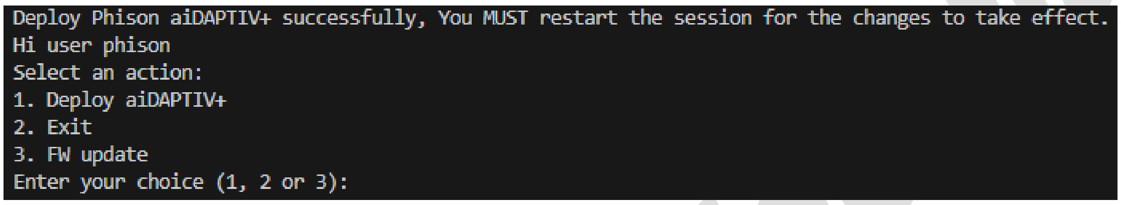
- If you only have one SSD, please follow the steps below to mount the drive:

```
sudo mkfs -t ext4 /dev/nvme1n1
sudo mkdir -p /mnt/nvme0
sudo mount /dev/nvme1n1 /mnt/nvme0
sudo chown -R $USER:$USER /mnt/nvme0
```

2.2.3. FW update

If fw update is not performed during installation, you can use setup_vNXUN_2_03_00.sh to do fw check/update.

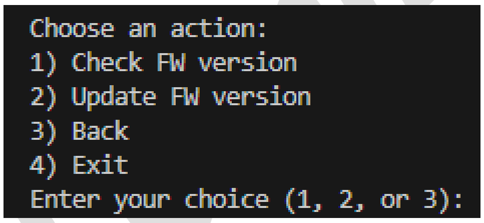
- If aiDaptiv+ is successfully installed is correct, a third option “FW update” will appear.Select “3. FW update”



```
Deploy Phison aiDAPTIV+ successfully, You MUST restart the session for the changes to take effect.
Hi user phison
Select an action:
1. Deploy aiDAPTIV+
2. Exit
3. FW update
Enter your choice (1, 2 or 3):
```

Figure 2-11 FW update

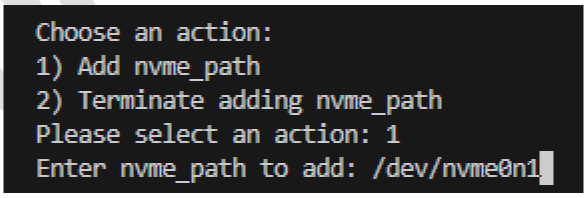
- You can choose to FW check/update



```
Choose an action:
1) Check FW version
2) Update FW version
3) Back
4) Exit
Enter your choice (1, 2, or 3):
```

Figure 2-12 FW check/update

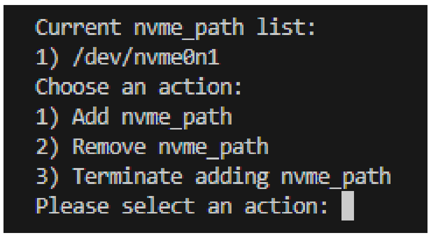
  - After selecting check/update, select 1 to add the device that needs to perform fw check/update



```
Choose an action:
1) Add nvme_path
2) Terminate adding nvme_path
Please select an action: 1
Enter nvme_path to add: /dev/nvme0n1
```

Figure 2-13 Select check/update FW

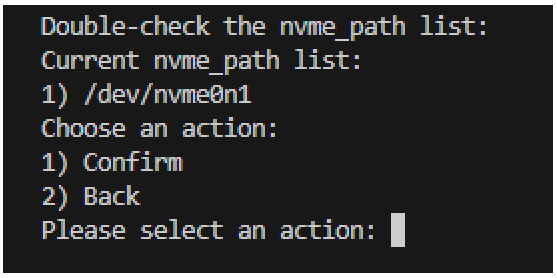
  - The added device will be displayed at the top. If you make a mistake in adding, select 2 to remove it. If you are done adding, select 3 to continue.



```
Current nvme_path list:
1) /dev/nvme0n1
Choose an action:
1) Add nvme_path
2) Remove nvme_path
3) Terminate adding nvme_path
Please select an action:
```

Figure 2-14 Add nvme_path

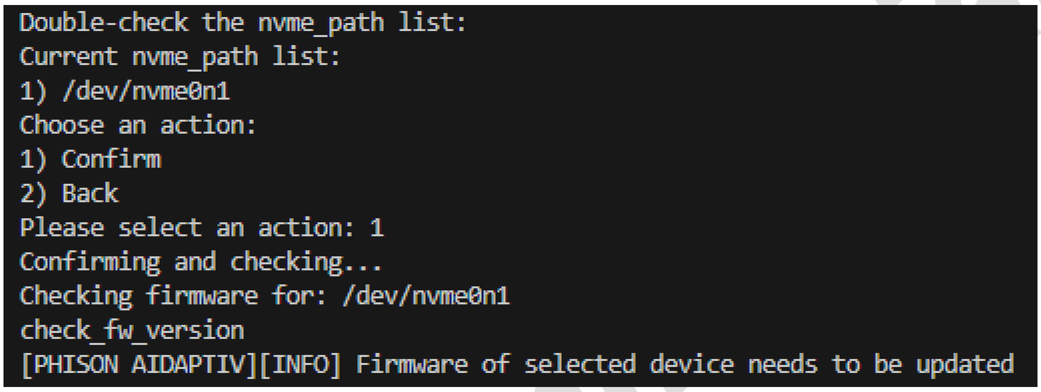
- Confirm again whether the added device is correct



```
Double-check the nvme_path list:
Current nvme_path list:
1) /dev/nvme0n1
Choose an action:
1) Confirm
2) Back
Please select an action: 
```

Figure 2-15 Repeat while more than one disk

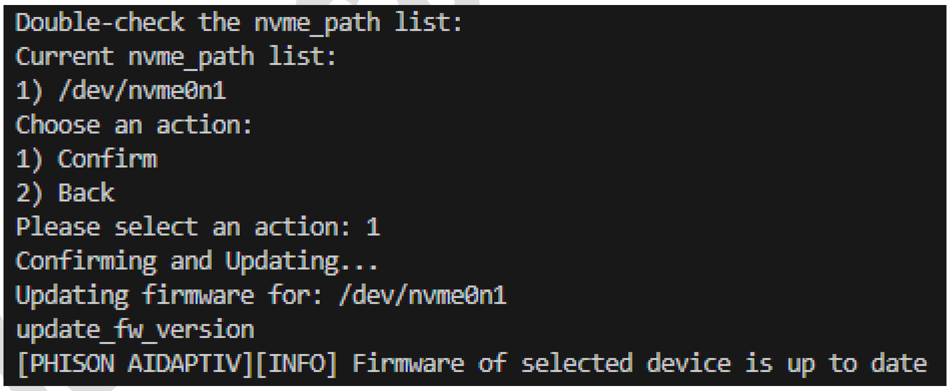
- Successful example
  - FW check



```
Double-check the nvme_path list:
Current nvme_path list:
1) /dev/nvme0n1
Choose an action:
1) Confirm
2) Back
Please select an action: 1
Confirming and checking...
Checking firmware for: /dev/nvme0n1
check_fw_version
[PHISON AIDAPTIV][INFO] Firmware of selected device needs to be updated
```

Figure 2-16 FW update check

  - FW update



```
Double-check the nvme_path list:
Current nvme_path list:
1) /dev/nvme0n1
Choose an action:
1) Confirm
2) Back
Please select an action: 1
Confirming and Updating...
Updating firmware for: /dev/nvme0n1
update_fw_version
[PHISON AIDAPTIV][INFO] Firmware of selected device is up to date
```

Figure 2-17 FW updated

2.3. Login Huggingface

2.3.1. How to get Hugging Face token

To download models from Hugging Face a personal token needs to be created and added to your account. You can obtain your personal token at: https://huggingface.co/settings/tokens

First, please register for a Hugging Face account. If you have already registered, please log in to Hugging Face directly.

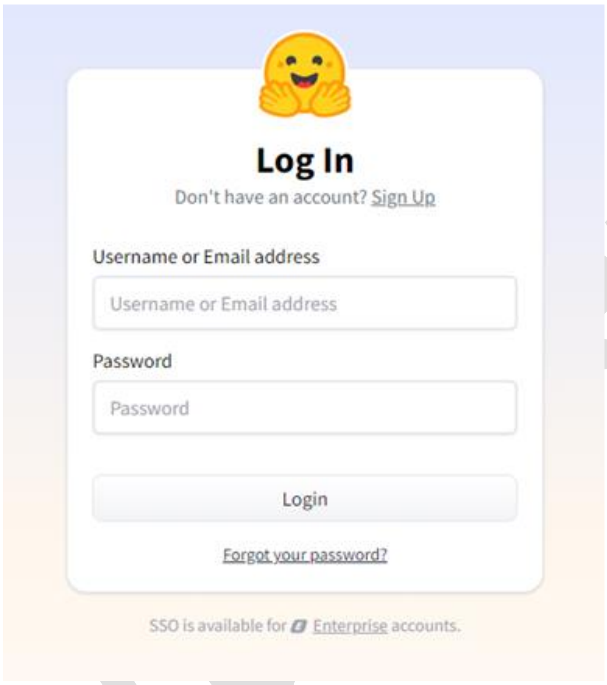

Log In
Don't have an account? Sign Up
Username or Email address
Username or Email address
Password
Password
Login
Forgot your password?
SSO is available for Enterprise accounts.

Figure 2-18 Login huggingface

After logging in, click on the account in the top right corner, open the menu, and select 'Settings'.

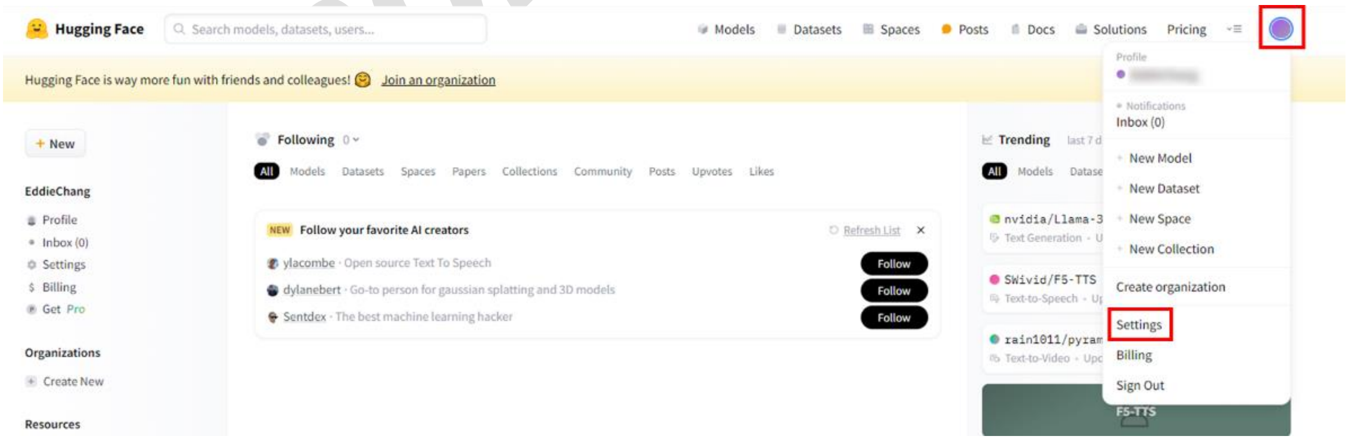
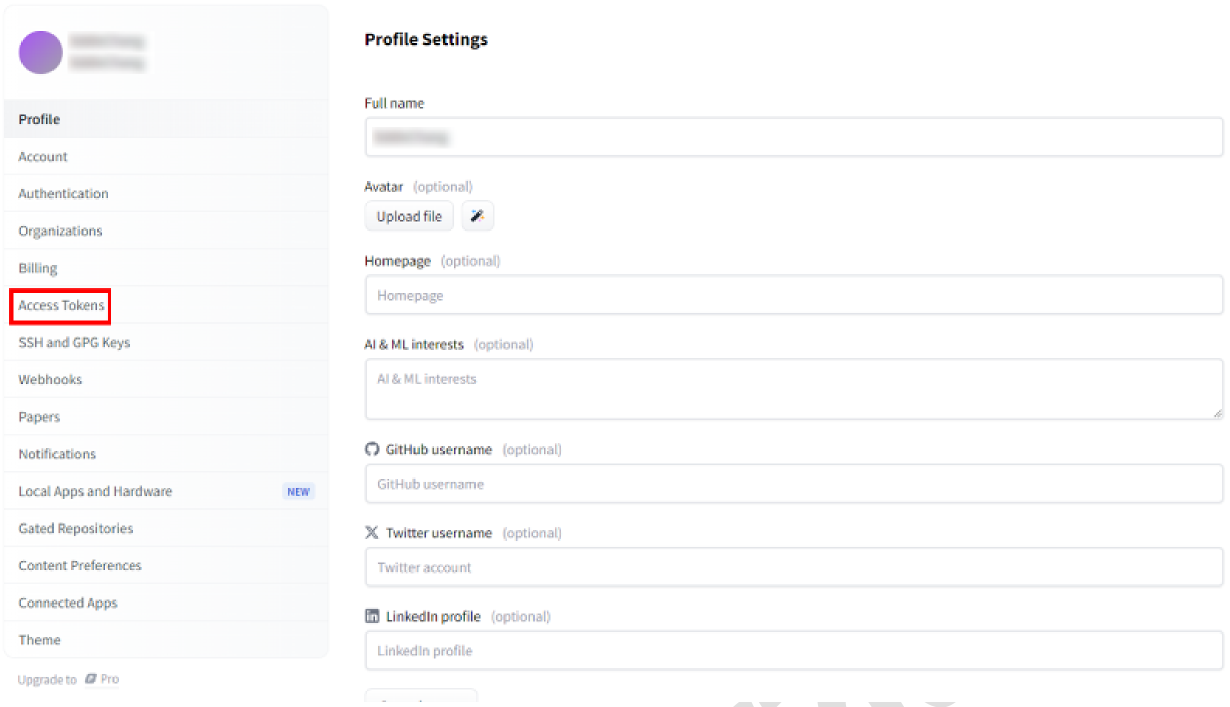


Figure 2-19 selec setting

After entering ‘Settings’, click on ‘Access Tokens’ in the left column.



Profile
Account
Authentication
Organizations
Billing
Access Tokens
SSH and GPG Keys
Webhooks
Papers
Notifications
Local Apps and Hardware NEW
Gated Repositories
Content Preferences
Connected Apps
Theme
Upgrade to Pro

Profile Settings

Full name

Avatar (optional)
Upload file

Homepage (optional)
Homepage

AI & ML interests (optional)
AI & ML interests

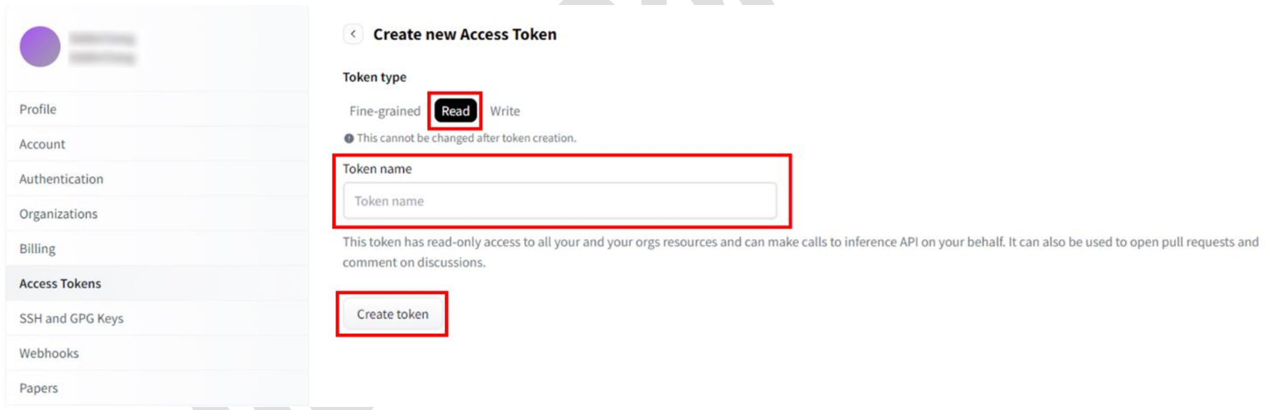
GitHub username (optional)
GitHub username

Twitter username (optional)
Twitter account

LinkedIn profile (optional)
LinkedIn profile

Figure 2-20 Access tokens

Choose the ‘Token type’ and enter the ‘Token name’, then you can create the required token. You can make a selection based on the content and description of the ‘Token type’. If you only need to download, choose ‘Read’.



Profile
Account
Authentication
Organizations
Billing
Access Tokens
SSH and GPG Keys
Webhooks
Papers

Create new Access Token

Token type
Fine-grained [✓ Read] Write
This cannot be changed after token creation.

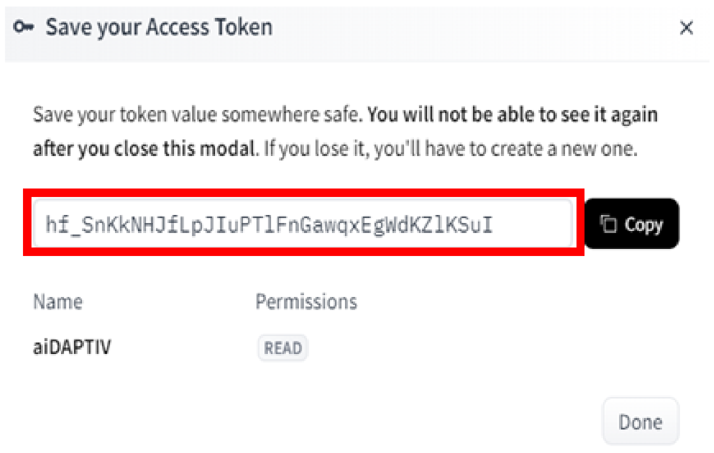
Token name
Token name

This token has read-only access to all your and your orgs resources and can make calls to inference API on your behalf. It can also be used to open pull requests and comment on discussions.

Create token

Figure 2-21 selec setting

After choosing to create, a Hugging Face token will appear. This token is used for logging in later.



Save your Access Token

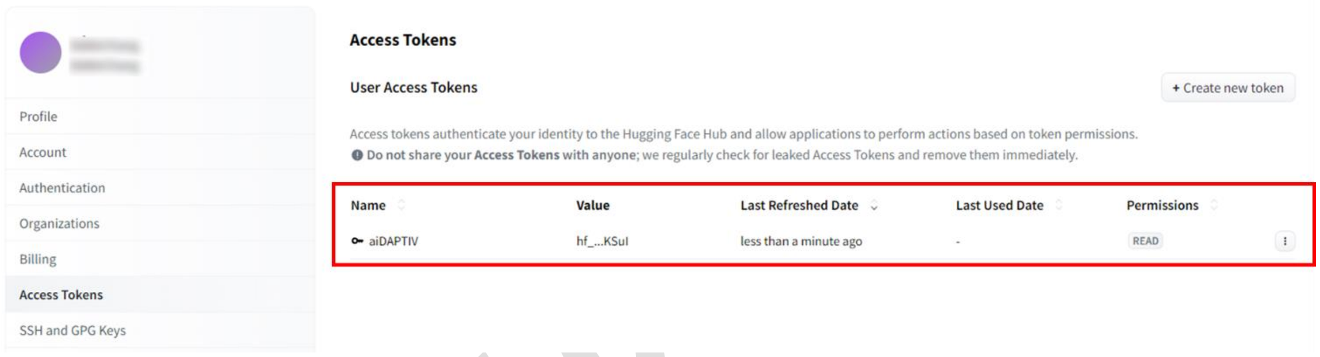
Save your token value somewhere safe. **You will not be able to see it again after you close this modal.** If you lose it, you'll have to create a new one.

```
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
```

Name	Permissions
aiDAPTIV	READ

Figure 2-22 Get tokens

The created token will be stored in the 'Access Token' in your personal account.



Access Tokens

User Access Tokens

Access tokens authenticate your identity to the Hugging Face Hub and allow applications to perform actions based on token permissions.
Do not share your Access Tokens with anyone; we regularly check for leaked Access Tokens and remove them immediately.

Name	Value	Last Refreshed Date	Last Used Date	Permissions
aiDAPTIV	hf_...KSuI	less than a minute ago	-	READ

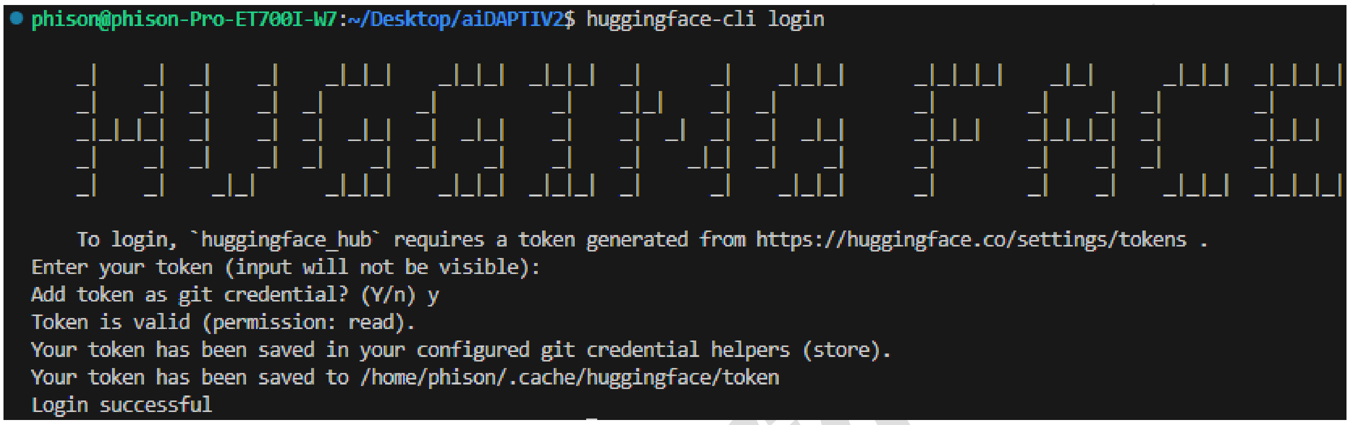
Figure 2-23 Get tokens in account

2.3.2. How to register the Hugging Face token

```
git config --global credential.helper store
huggingface-cli login
# login with <your_hf_token>
```

- Successful example

There would be a “Login successful” message.

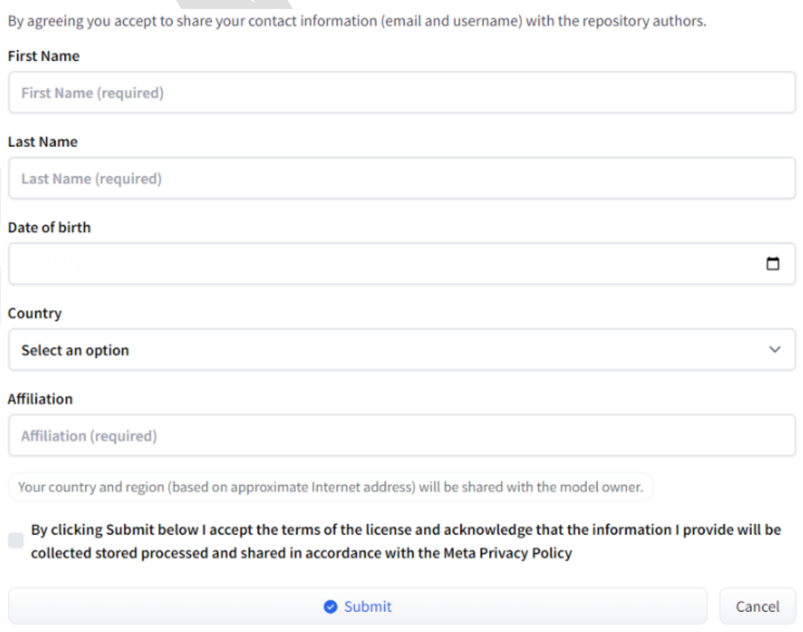


```
phison@phison-Pro-ET700I-W7:~/Desktop/aiDAPTIV2$ huggingface-cli login

    _|    _|  _|    _|    _|_|_|    _|_|_|  _|_|_|  _|      _|    _|_|_|      _|_|_|_|    _|_|      _|_|_|  _|_|_|_|
    _|    _|  _|    _|  _|        _|          _|    _|_|    _|  _|            _|        _|    _|  _|        _|
    _|_|_|_|  _|    _|  _|  _|_|  _|  _|_|    _|    _|  _|  _|  _|  _|_|      _|_|_|    _|_|_|_|  _|        _|_|_|
    _|    _|  _|    _|  _|    _|  _|    _|    _|    _|    _|_|  _|    _|      _|        _|    _|  _|        _|
    _|    _|    _|_|      _|_|_|    _|_|_|  _|_|_|  _|      _|    _|_|_|      _|        _|    _|    _|_|_|  _|_|_|_|

    To login, `huggingface_hub` requires a token generated from https://huggingface.co/settings/tokens .
Enter your token (input will not be visible):
Add token as git credential? (Y/n) y
Token is valid (permission: read).
Your token has been saved in your configured git credential helpers (store).
Your token has been saved to /home/phison/.cache/huggingface/token
Login successful
```

Figure 2-24 Login huggingface

2.4. Download Llama-3.1-8B-Instruct

- Before downloading the Llama-3.1-8B-Instruct model, you need to request permission from huggingface.
  - https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct
  - It needs to wait for the license approval from huggingface



By agreeing you accept to share your contact information (email and username) with the repository authors.

First Name

First Name (required)

Last Name

Last Name (required)

Date of birth

Country

Select an option

Affiliation

Affiliation (required)

Your country and region (based on approximate Internet address) will be shared with the model owner.

By clicking Submit below I accept the terms of the license and acknowledge that the information I provide will be collected stored processed and shared in accordance with the Meta Privacy Policy

Figure 2-25 Download Llama-3.1-8B-Instruct

2.4.1. Download Llama-3.1-8B-Instruct model

```
mkdir -p /home/$USER/Desktop/llm
cd /home/$USER/Desktop/llm
mkdir Llama-3.1-8B-Instruct
huggingface-cli download --token HF_TOKEN --resume-download meta-llama/Llama-3.1-8B-Instruct \
--local-dir-use-symlinks False --local-dir Llama-3.1-8B-Instruct
# It would be take some time to download model.
```

HF_TOKEN: Please replace with <your_hf_token>
meta-llama/Llama-3.1-8B-Instruct: You can replace it with the model you want to download.

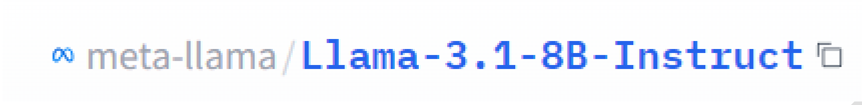
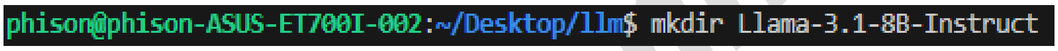

∞ meta-llama/Llama-3.1-8B-Instruct

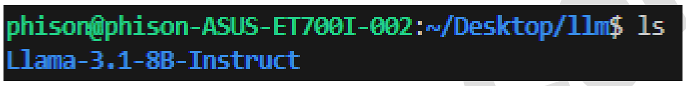
Figure 2-26 Model Name

Llama-3.1-8B-Instruct: Please replace it with the folder you created

- Download success message
  - mkdir -p Llama-3.1-8B-Instruct in /home/$USER/Desktop/llm/

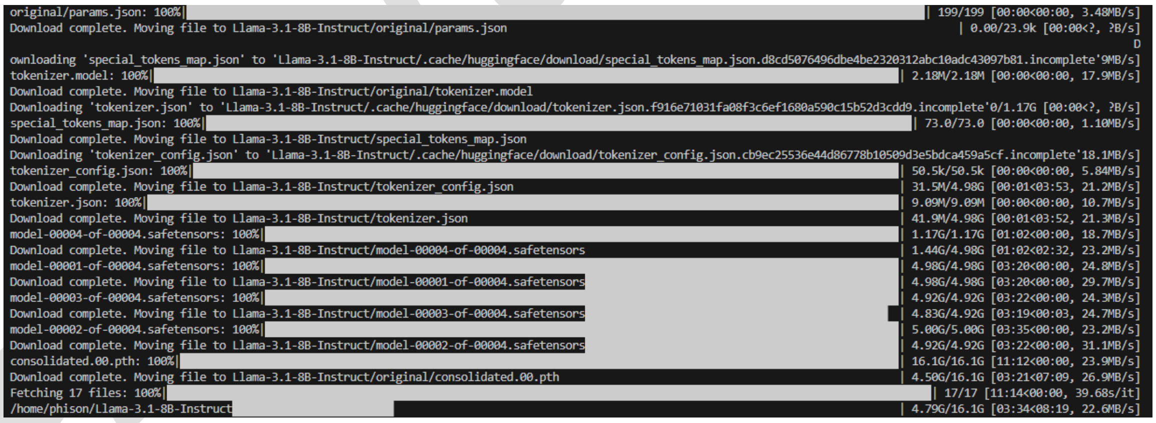


```
phison@phison-ASUS-ET700I-002:~/Desktop/llm$ mkdir Llama-3.1-8B-Instruct
```



```
phison@phison-ASUS-ET700I-002:~/Desktop/llm$ ls
Llama-3.1-8B-Instruct
```

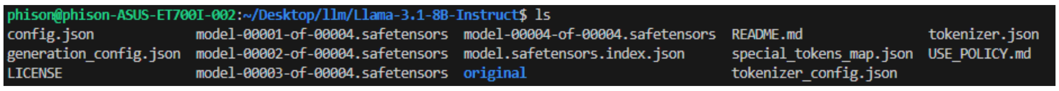
  - Download success message



```
original/params.json: 100%|                                   | 199/199 [00:00<00:00, 3.48MB/s]
Download complete. Moving file to Llama-3.1-8B-Instruct/original/params.json    | 0.00/23.9k [00:00<?, ?B/s]
                                                                                                  D
ownloading 'special_tokens_map.json' to 'Llama-3.1-8B-Instruct/.cache/huggingface/download/special_tokens_map.json.d8cd5076496dbe4be2320312abc10adc43097b81.incomplete'9MB/s]
tokenizer.model: 100%|                                  | 2.18M/2.18M [00:00<00:00, 17.9MB/s]
Download complete. Moving file to Llama-3.1-8B-Instruct/original/tokenizer.model
Downloading 'tokenizer.json' to 'Llama-3.1-8B-Instruct/.cache/huggingface/download/tokenizer.json.f916e71031fa08f3c6ef1680a590c15b52d3cdd9.incomplete'0/1.17G [00:00<?, ?B/s]
special_tokens_map.json: 100%|                          | 73.0/73.0 [00:00<00:00, 1.10MB/s]
Download complete. Moving file to Llama-3.1-8B-Instruct/special_tokens_map.json
Downloading 'tokenizer_config.json' to 'Llama-3.1-8B-Instruct/.cache/huggingface/download/tokenizer_config.json.cb9ec25536e44d86778b10509d3e5bdca459a5cf.incomplete'18.1MB/s]
tokenizer_config.json: 100%|                            | 50.5k/50.5k [00:00<00:00, 5.84MB/s]
Download complete. Moving file to Llama-3.1-8B-Instruct/tokenizer_config.json    | 31.5M/4.98G [00:01<03:53, 21.2MB/s]
tokenizer.json: 100%|                                   | 9.09M/9.09M [00:00<00:00, 10.7MB/s]
Download complete. Moving file to Llama-3.1-8B-Instruct/tokenizer.json           | 41.9M/4.98G [00:01<03:52, 21.3MB/s]
model-00004-of-00004.safetensors: 100%|                 | 1.17G/1.17G [01:02<00:00, 18.7MB/s]
Download complete. Moving file to Llama-3.1-8B-Instruct/model-00004-of-00004.safetensors | 1.44G/4.98G [01:02<02:32, 23.2MB/s]
model-00001-of-00004.safetensors: 100%|                 | 4.98G/4.98G [03:20<00:00, 24.8MB/s]
Download complete. Moving file to Llama-3.1-8B-Instruct/model-00001-of-00004.safetensors | 4.98G/4.98G [03:20<00:00, 29.7MB/s]
model-00003-of-00004.safetensors: 100%|                 | 4.92G/4.92G [03:22<00:00, 24.3MB/s]
Download complete. Moving file to Llama-3.1-8B-Instruct/model-00003-of-00004.safetensors | 4.83G/4.92G [03:19<00:03, 24.7MB/s]
model-00002-of-00004.safetensors: 100%|                 | 5.00G/5.00G [03:35<00:00, 23.2MB/s]
Download complete. Moving file to Llama-3.1-8B-Instruct/model-00002-of-00004.safetensors | 4.92G/4.92G [03:22<00:00, 31.1MB/s]
consolidated.00.pth: 100%|                              | 16.1G/16.1G [11:12<00:00, 23.9MB/s]
Download complete. Moving file to Llama-3.1-8B-Instruct/original/consolidated.00.pth | 4.50G/16.1G [03:21<07:09, 26.9MB/s]
Fetching 17 files: 100%|                                 | 17/17 [11:14<00:00, 39.68s/it]
/home/phison/Llama-3.1-8B-Instruct                       | 4.79G/16.1G [03:34<08:19, 22.6MB/s]
```

Figure 2-27 Download Llama-3.1-8B-Instruct successful

  - The weight of the model will appear in "Llama-3.1-8B-Instruct"



```
phison@phison-ASUS-ET700I-002:~/Desktop/llm/Llama-3.1-8B-Instruct$ ls
config.json             model-00001-of-00004.safetensors  model-00004-of-00004.safetensors  README.md                tokenizer.json
generation_config.json  model-00002-of-00004.safetensors  model.safetensors.index.json      special_tokens_map.json  USE_POLICY.md
LICENSE                 model-00003-of-00004.safetensors  original                          tokenizer_config.json
```

Figure 2-28 Weight of the model

2.5. Run aiDAPTIV

The command launcher for aiDAPTIV is “phisonai2” and will be used to start a training session.

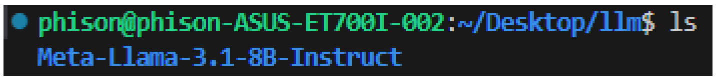
Note: It can take several hours to finish the job depending the size of your dataset and system performance.

2.5.1. Start training your model

Training model “Llama-3.1-8B-Instruct” with dataset “Dahoas/rm-static”

```
phisonai2 --env_config <env_config.yaml path> --exp_config <exp_config.yaml path>
```

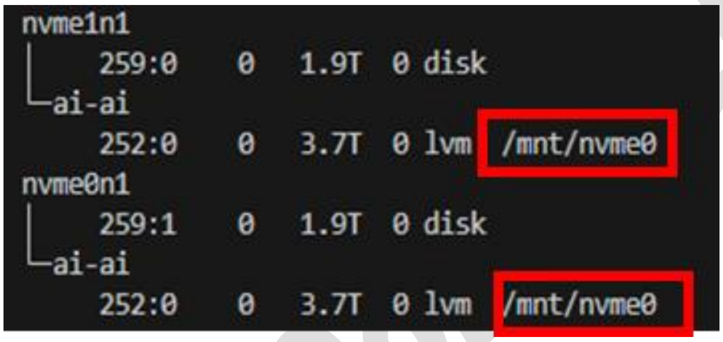
First, find the location where the model is stored.



```
phison@phison-ASUS-ET700I-002:~/Desktop/llm$ ls
Meta-Llama-3.1-8B-Instruct
```

Figure 2-29 Location of model

Use the ‘lsblk’ command to confirm the location of the mounted SSD.



```
nvme1n1
│    259:0    0   1.9T  0 disk
└ai-ai
     252:0    0   3.7T  0 lvm  /mnt/nvme0
nvme0n1
│    259:1    0   1.9T  0 disk
└ai-ai
     252:0    0   3.7T  0 lvm  /mnt/nvme0
```

Figure 2-30 Location of SSD mount path

Write the required parameters into

‘/home/$USER/aiDAPTIV2/commands/env_config/env_config.yaml’

‘/home/$USER/aiDAPTIV2/commands/exp_config/exp_config.yaml’

or replace it with the path where you store ‘env_config.yaml’ and ‘exp_config.yaml’.

Remark:

1. Remember to replace $USER with the current user.
2. For more efficient training, it is recommended to set ‘triton’ to true.

2.5.2. text-generation settings

- Example of env_config.yaml.

```
# Save_path, nvme_path, log_name settings
path_settings:
  lora:
    lora_weight: ""       # whether to load lora_weight (only activated when lora: true)
    lora_output_dir: ""   # whether to save lora adapter weight (only activated when lora: true)
  model_name_or_path: "/home/$USER/Desktop/llm/Llama-3.1-8B-Instruct"
  data_path:
    # - ./dataset_config/image-text-to-text/VQA_dataset_config.yaml
    # - ./dataset_config/text-generation/Pretrain_dataset_config.yaml
    # - ./dataset_config/text-generation/RAG_dataset_config.yaml
    - ./dataset_config/text-generation/QA_dataset_config.yaml
  nvme_path: "/mnt/nvme0"
  output_dir: "/home/$USER/output"
  log_name: "Llama-3.1-8B-Instruct.log"
```

- Example of exp_config.yaml.

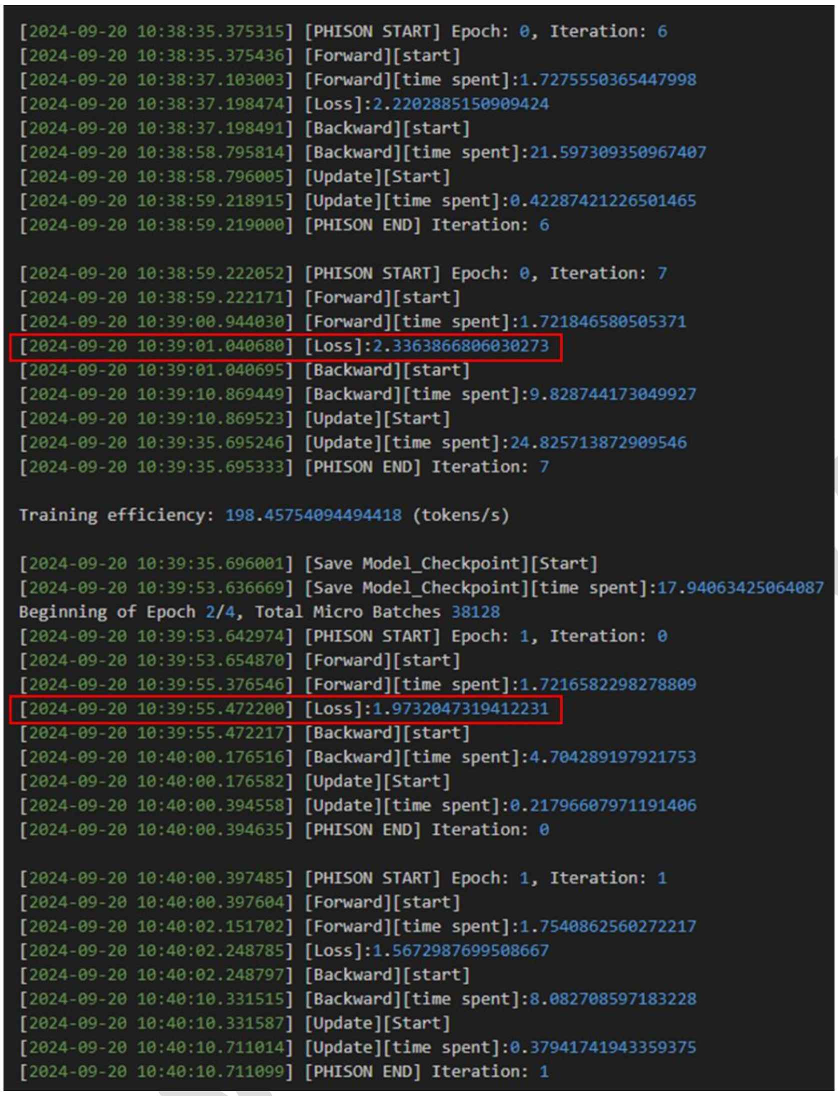
```
process_settings:
  master_port: 8299
  num_gpus: 1
  specify_gpus: null
run_settings:
  task_type: "text-generation"
  task_mode: "train" # or "/home/$USER/aiDAPTIV2/text-generation/train.py"
  per_device_train_batch_size: 1
  per_update_total_batch_size: 4
  num_train_epochs: 1
  max_iter: -1
  max_seq_len: 2048
  triton: True
  weight_file_format: null
  from_config: false
  precision_mode: 1

 model_saver:
    max_num_of_saved_model_on_epoch_end: -1 # If the value is -1, it is equal to num_train_epochs.
    enable_save_model_on_iteration: false
    max_num_of_saved_model_on_iteration: 2
    num_of_iteration_to_save_model: 2

  lr_scheduler:
    mode: -1
    learning_rate: 0.000007

  optimizer:
    beta1: 0.9
    beta2: 0.95
    eps: 0.00000001
    weight_decay: 0.01

  lora:
    enable_lora: false
    lora_rank: 8
    lora_alpha: 16
    lora_task_type: "CAUSAL_LM"
```

2.5.3. Monitor training status

Open another window, the log will be generated inside the folder where the command is executed.

```
#Open another session
tail -f <your_log>
# example: tail -f Phison_2024_05-15_1200.log
```

- If the “Loss” value in log shows a decreasing trend the training was successful.



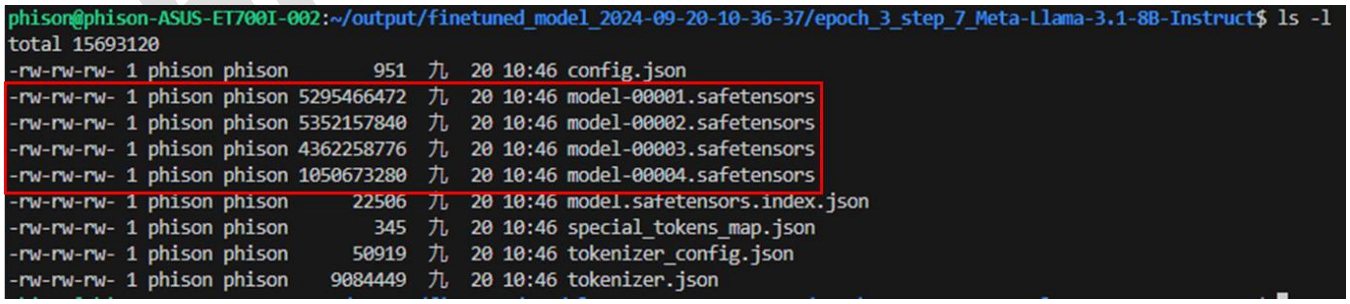
```
[2024-09-20 10:38:35.375315] [PHISON START] Epoch: 0, Iteration: 6
[2024-09-20 10:38:35.375436] [Forward][start]
[2024-09-20 10:38:37.103003] [Forward][time spent]:1.7275550365447998
[2024-09-20 10:38:37.198474] [Loss]:2.2202885150909424
[2024-09-20 10:38:37.198491] [Backward][start]
[2024-09-20 10:38:58.795814] [Backward][time spent]:21.597309350967407
[2024-09-20 10:38:58.796005] [Update][Start]
[2024-09-20 10:38:59.218915] [Update][time spent]:0.42287421226501465
[2024-09-20 10:38:59.219000] [PHISON END] Iteration: 6

[2024-09-20 10:38:59.222052] [PHISON START] Epoch: 0, Iteration: 7
[2024-09-20 10:38:59.222171] [Forward][start]
[2024-09-20 10:39:00.944030] [Forward][time spent]:1.721846580505371
[2024-09-20 10:39:01.040680] [Loss]:2.3363866806030273
[2024-09-20 10:39:01.040695] [Backward][start]
[2024-09-20 10:39:10.869449] [Backward][time spent]:9.828744173049927
[2024-09-20 10:39:10.869523] [Update][Start]
[2024-09-20 10:39:35.695246] [Update][time spent]:24.825713872909546
[2024-09-20 10:39:35.695333] [PHISON END] Iteration: 7

Training efficiency: 198.45754094494418 (tokens/s)

[2024-09-20 10:39:35.696001] [Save Model_Checkpoint][Start]
[2024-09-20 10:39:53.636669] [Save Model_Checkpoint][time spent]:17.94063425064087
Beginning of Epoch 2/4, Total Micro Batches 38128
[2024-09-20 10:39:53.642974] [PHISON START] Epoch: 1, Iteration: 0
[2024-09-20 10:39:53.654870] [Forward][start]
[2024-09-20 10:39:55.376546] [Forward][time spent]:1.7216582298278809
[2024-09-20 10:39:55.472200] [Loss]:1.9732047319412231
[2024-09-20 10:39:55.472217] [Backward][start]
[2024-09-20 10:40:00.176516] [Backward][time spent]:4.704289197921753
[2024-09-20 10:40:00.176582] [Update][Start]
[2024-09-20 10:40:00.394558] [Update][time spent]:0.21796607971191406
[2024-09-20 10:40:00.394635] [PHISON END] Iteration: 0

[2024-09-20 10:40:00.397485] [PHISON START] Epoch: 1, Iteration: 1
[2024-09-20 10:40:00.397604] [Forward][start]
[2024-09-20 10:40:02.151702] [Forward][time spent]:1.7540862560272217
[2024-09-20 10:40:02.248785] [Loss]:1.5672987699508667
[2024-09-20 10:40:02.248797] [Backward][start]
[2024-09-20 10:40:10.331515] [Backward][time spent]:8.082708597183228
[2024-09-20 10:40:10.331587] [Update][Start]
[2024-09-20 10:40:10.711014] [Update][time spent]:0.37941741943359375
[2024-09-20 10:40:10.711099] [PHISON END] Iteration: 1
```

Figure 2-31 aiDAPTIV training log

2.5.4. *Successful example*

You get your first fine-tuned model in <output_dir> /home/$USER/output !!

Fine-tuned model file: model-index.safetensors



```
phison@phison-ASUS-ET700I-002:~/output/finetuned_model_2024-09-20-10-36-37/epoch_3_step_7_Meta-Llama-3.1-8B-Instruct$ ls -l
total 15693120
-rw-rw-rw- 1 phison phison        951 九  20 10:46 config.json
-rw-rw-rw- 1 phison phison 5295466472 九  20 10:46 model-00001.safetensors
-rw-rw-rw- 1 phison phison 5352157840 九  20 10:46 model-00002.safetensors
-rw-rw-rw- 1 phison phison 4362258776 九  20 10:46 model-00003.safetensors
-rw-rw-rw- 1 phison phison 1050673280 九  20 10:46 model-00004.safetensors
-rw-rw-rw- 1 phison phison      22506 九  20 10:46 model.safetensors.index.json
-rw-rw-rw- 1 phison phison        345 九  20 10:46 special_tokens_map.json
-rw-rw-rw- 1 phison phison      50919 九  20 10:46 tokenizer_config.json
-rw-rw-rw- 1 phison phison    9084449 九  20 10:46 tokenizer.json
```

Figure 2-32 Finetuned model

2.5.5. *image-text-to-text settings*

- Example of env_config.yaml.

```
# Save_path, nvme_path, log_name settings
path_settings:
  lora:
    lora_weight: ""       # whether to load lora_weight (only activated when lora: 
true)
    lora_optimizer: ""       # whether to load lora_optimizer (only activated when 
lora: true)
    lora_output_dir: ""   # whether to save lora adapter weight (only activated when 
lora: true)
  model_name_or_path: "/home/$USER/Desktop/llm/Llama-3.2-11B-Vision-Instruct"
  multi_node_env_path: null
  optimizer_path: "" # whether to load optimizer
  train_data_path:
    # - ./dataset_config/automatic-speech-recognition/dataset_config.yaml
    - ./dataset_config/image-text-to-text/VQA_dataset_config.yaml
    # - ./dataset_config/text-generation/Pretrain_dataset_config.yaml
    # - ./dataset_config/text-generation/RAG_dataset_config.yaml
    # - ./dataset_config/text-generation/QA_dataset_config.yaml
  val_data_path: null
    # - ./dataset_config/automatic-speech-recognition/dataset_config.yaml
    # - ./dataset_config/image-text-to-text/VQA_dataset_config.yaml
    # - ./dataset_config/text-generation/Pretrain_dataset_config.yaml
    # - ./dataset_config/text-generation/RAG_dataset_config.yaml
    # - ./dataset_config/text-generation/QA_dataset_config.yaml
  nvme_path: "/mnt/nvme0"
  output_dir: "/home/$USER/output"
  log_name: "Llama-3.2-11B-Vision-Instruct.log"
```

- Example of exp_config.yaml

```
process_settings:
 master_port: 8299
 num_gpus: 1
 specify_gpus: null
 master_port: 8299
 multi_node_settings:
     enable: False
     master_addr: "127.0.0.1"

run_settings:
 task_type: "image-text-to-text"
 task_mode: "train" # or "/home/$USER/aiDAPTIV2/image-text-to-text/train.py"
 per_device_train_batch_size: 1
 per_update_total_batch_size: 4
 num_train_epochs: 1
 max_iter: -1
 max_seq_len: 2048
 triton: True
 weight_file_format: null
 from_config: false
 precision_mode: 1
 enable_save_optimizer_state: false

model_saver:
   max_num_of_saved_model_on_epoch_end: -1 # If the value is -1, it is equal to
num_train_epochs.
   enable_save_model_on_iteration: false
   max_num_of_saved_model_on_iteration: 2
   num_of_iteration_to_save_model: 2

 lr_scheduler:
   mode: -1
   learning_rate: 0.000007

 optimizer:
   beta1: 0.9
   beta2: 0.95
   eps: 0.00000001
   weight_decay: 0.01

 early_stop:
   enable: false
   min_delta: 0.01
   patience: 2
   verbose: false
 lora:
   enable_lora: false
   lora_rank: 8
   lora_alpha: 16
   lora_task_type: "CAUSAL_LM"
   lora_target_modules: null
```

2.5.6. "phisonai2" command arguments

You can complete a training task using the following command.

[Warning] You cannot enable *Triton* and *Lora* simultaneously!

- env_config.yaml

```
path_settings:
  lora:
    lora_weight: "str"       # whether to load lora_weight (only activated when lora: true)
    lora_optimizer: "str"       # whether to load lora_optimizer (only activated when lora: true)
    lora_output_dir: "str"   # whether to save lora adapter weight (only activated when lora: true)
  model_name_or_path: "str"
  multi_node_env_path: null
  optimizer_path: "str" # whether to load optimizer
  train_data_path:
    # - ./dataset_config/image-text-to-text/VQA_dataset_config.yaml
    # - ./dataset_config/text-generation/Pretrain_dataset_config.yaml
    # - ./dataset_config/text-generation/RAG_dataset_config.yaml
    - ./dataset_config/text-generation/QA_dataset_config.yaml
  val_data_path: null
    # - ./dataset_config/image-text-to-text/VQA_dataset_config.yaml
    # - ./dataset_config/text-generation/Pretrain_dataset_config.yaml
    # - ./dataset_config/text-generation/RAG_dataset_config.yaml
    # - ./dataset_config/text-generation/QA_dataset_config.yaml
  nvme_path: "str"
  output_dir: "str"
  log_name: "str"
```

- exp_config.yaml

```
process_settings:                              # <examples>
    num_gpus: 'int'                            # 1, 2, 4, 8
    specify_gpus: [null|'str']                 # null, '0,1,2,3', '4,5'
    master_port: 'int'                         # 8299
    multi_node_settings:
        enable: 'bool'                         # true, false
        master_addr: 'str'                     # "127.0.0.1"

run_settings:
    task_type 'str'                            # 'text-generation', 'automatic-speech-
                                               recognition', 'fill-mask', 'image-text-
                                               to-text'
    task_mode 'str'                            # 'train', or '/home/$USER/
                                               aiDAPTIV2/text-generation/train.py'


    per_device_train_batch_size 'int'          # 10
    per_update_total_batch_size 'int'          # 100 It must be set as a multiple of
                                               (num_gpus * per_device_train_batch_siz
                                               e).
    num_train_epochs 'int'                     # 5
    max_iter 'int'                             # -1, 100
    max_seq_len 'int'                          # 2048
    triton 'bool'                              # true, false
    weight_file_format [null|'str']            # null, 'bin', 'pt', 'safetensors'
    from_config 'bool'                         # true, false
    precision_mode 'int'                       # 0, 1
    enable_save_optimizer_state: 'bool'        # true, false

model_saver:
    max_num_of_saved_model_on_epoch_end: -1    # If the value is -1, it is equal to
    enable_save_model_on_iteration: false      num_train_epochs.
    max_num_of_saved_model_on_iteration: 2
    num_of_iteration_to_save_model: 2

lr_scheduler
    mode 'int'                                 # -1, 0, 1, 2, 3, 4, 5
    learning_rate 'float'                      # 0.000007

optimizer
    beta1 'float'                              # 0.9
    beta2 'float'                              # 0.95
    eps 'float'                                # 0.00000001
    weight_decay 'float'                       # 0.01
```

```
early_stop:
  enable: 'bool'                                    # true, false
  min_delta: 'float'                                # 0.01
  patience: 'int'                                   # 2
  verbose: 'bool'                                   # true, false


lora
    enable_lora 'bool'                              # true, false
    lora_rank 'int'                                 # 8
    lora_alpha 'int'                                # 16
    lora_task_type 'str'                            # 'CAUSAL_LM'
    lora_target_modules: null,'str',list[st         # null
    r]
```

Arguments Descriptions

- **path_setting**:
  - **Lora:**
    - **lora_weight**: absolute path to the pretrained lora weight folder(**default: None**).
    - **lora_optimizer**: absolute path to the lora optimizer (**default: None**).
    - **lora_output_dir**: absolute path for saving lora model (default: None, lora model will only be saved if you provide lora_output_dir). (**default: None**).
  - **model_name_or_path**: **[Necessary!]** absolute path to the pretrained weight folder (downloaded from huggingface) (**default: None**).
  - **multi_node_env_path**: Multi node env config (**default: Null**).
  - **optimizer_path**: Input the path of the optimizer state saved from the previous training session. The current training session will continue based on this optimizer state. (**default: None**).
  - **train_data_path**: When adding dataset, the yaml file under **/commands/env_configs** need to be configured.
    - Currently only support :

```
- VQA datasets
- RAG datasets
- QA datasets
- Pretrain datasets
```

For RAG, QA dataset, **single** QA pair or **multi-turn** chat can both be accepted.

After configured, the yaml file path should be added to **commands/env_config/env_config.yaml**

```
train_data_path:
   - ./dataset_config/image-text-to-text/VQA_dataset_config.yaml
   - ./dataset_config/text-generation/Pretrain_dataset_config.yaml
   - ./dataset_config/text-generation/RAG_dataset_config.yaml
   - ./dataset_config/text-generation/QA_dataset_config.yaml
```

***To run the default env_config.yaml, the execute path should locate at aiDAPTIV2/commands.**

  - **val_data_path**: absolute path to the dataset used to validate the training (**default: None**).

  - **nvme_path**: **[Necessary!]** mounting point to aiDAPTIVCache (ex:/mnt/nvme0) (**default: None**).
  - **output_dir**: absolute path for saving finetuned model (**default: None, finetuned model will only be saved if you provide output_dir**).
  - **log_name**: absolute path for saving training log (**default: Phison_"currenttime".log**).
- **process_settings**:
  - **num_gpus**: number of gpus to be utilized (**default: 1**).
  - **specify_gpus**: (optional) specify GPUs to use. If you want to use No3 and No2 GPU, set specify_gpus="3,2" (**default: null**).
  - **master_port**: port used by PyTorch distributed for communication during training (**default: 8299**).
  - **multi_node_settings**:
    - **enable**: trigger multi node training (**default: False**).
    - **master_addr**: Master node (rank 0)'s address, should be either the IP address or the hostname of node 0, for single node multi-proc training, the --master_addr can simply be 127.0.0.1 (**default: 127.0.0.1**).
- **run_settings**:
  - **task_type**: Choose a task type, the folder in /home/$USER/aiDAPTIV2. We only support "text_generation", "automatic-speech-recognition" and "fill-mask" in this version (**default: text_generation**). Check task type and model are in AVL (Appendix D Support Model list).
  - **task_mode**: Choose a task mode. We support "train", "inference", "eval", and absolute path to the execution python file, ex: "/home/$USER/aiDAPTIV2/text-generation/train.py" (**default: train**).
  - **per_device_train_batch_size**: batch size in each GPU (**default: 1**).
  - **per_update_total_batch_size**: batch size for one update (**default: 128**).

    remark : It must be set as a multiple of (num_gpus * per_device_train_batch_size).

    Ex: if you have 4 GPUs, each of them have 4 batches. you want to update the model every 80 batches. Then set the per_device_train_batch_size = 4, per_update_total_batch_size = 80. Your machine will run 80/4/4=5 iterations and update once.
  - **num_train_epochs**: how many epoch you want to train your model (**default: 1**).
  - **max_iter**: (optional) early stop iteration before running entire epoch. Set -1: execute all iterations for each epoch, set 10: only execute 10 iterations for each epoch. (**default: -1**).
  - **max_seq_len**: sequence length you want to train your language model (**default: 2048**).
    - Model limitations: if you run bert model, you must set max_seq_len=512. For other models, please refer to the info on the huggingface's model page.
  - **triton**: (optional) trigger triton training procedure. We include some of techniques from Triton to improve training efficiency. Llama, Mistral, Mixtral are currently supported (**default: False**).
  - **weight_file_format**: file format for loading pretrained model. We only support "safetensors", "bin", "pt", "pth" and "None" in this version. (**default: None, we will automaticlly search file format**).
  - **from_config**: whether randomly init model weight. (**default: False**)
  - **precision_mode**: determines what precision to train your model. 0 for "BF16" and 1 for "BF16, FP32 mixed". (**default: 1**)

- **enable_save_optimizer_state**: Default is false, Indicates whether to save the current optimizer state to the system disk. If set to true, it means you want to save the optimizer state from this training session for use in the next training session. (**default: False**).
- **model_saver**
  - **max_num_of_saved_model_on_epoch_end**: Specifies the maximum number of models to be saved during epoch cycles. If the number of saved models exceeds this value during the training process, the first model saved in the epoch will be removed. Setting this value to -1 is equivalent to num_train_epochs, meaning that every model saved per epoch will be retained (**default: -1**).
  - **enable_save_model_on_iteration**: Specifies whether to save the model during iteration cycles (**default: false**).
  - **max_num_of_saved_model_on_iteration:** Specifies the maximum number of models to be saved during iteration cycles. The concept is similar to max_num_of_saved_model_on_epoch_end. The value must be greater than 0, and it is recommended to use at least 2 to avoid issues where models may be lost due to device problems during the saving process (**default: 2**).
  - **num_of_iteration_to_save_model**: Specifies how often a model should be saved per number of steps, and it must be a multiple of "args.gradient_accumulation_steps". The value must be greater than 0 (**default: 2**).
- **lr_scheduler**
  - **learning_rate**: learning rate, be aware of this hyper-parameter. It can affect training result (**default: 7e-6**).
  - **mode**: choose a learning rate mode (**default: 1**).
    - mode == -1: LinearLR (optimizer, start_factor=1, total_iters=1)
    - mode == 0: LinearLR (optimizer, start_factor=0.5, total_iters=20)
    - mode == 1: CosineAnnealingLR (optimizer, T_max=150)
    - mode == 2: ExponentialLR (optimizer, gamma=0.99)
    - mode == 3: MultiplicativeLR (optimizer, lr_lambda=lambda epoch: 0.95)
    - mode == 4: StepLR (optimizer, step_size=30, gamma=0.1)
    - mode == 5: MultiStepLR (optimizer, milestones=[30,80], gamma=0.1)
- **optimizer**
  - **beta1**: hyper-parameter for adam optimizer (**default: 0.9**).
  - **beta2**: hyper-parameter for adam optimizer (**default: 0.95**).
  - **eps**: hyper-parameter for adam optimizer (**default:1e-8**).
  - **weight_decay**: weight decay coefficient.(**default:1e-2**).
- **early_stop**
  - **enable**: trigger early stop (**default: False**).
  - **min_delta**: The current val_loss must be more than min_delta away from the best val_loss to be considered improved (**default: 0.01**).
  - **patience**: If val_loss does not improve and exceeds the patience times, early stopping will be triggered (**default:2**).
  - **verbose**: trigger to print loss.(**default:False**).

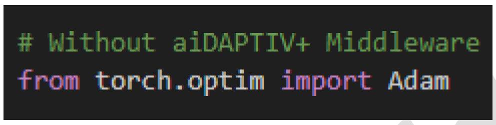
- **lora**
  - **enable_lora**: trigger lora training procedure (**default: False**).
  - **lora_rank**: dimension of the low-rank matrices for lora (**default: 8**).
  - **lora_alpha**: scaling factor of the weight matrices for lora (**default: 16**).
  - **lora_task_type**: training task type for lora. We only support “CAUSAL_LM” in this version (**default: “CAUSAL_LM”**).
  - **lora_target_modules:** The names of the modules to apply the adapter to. If this is specified, only the modules with the specified names will be replaced **(default: “null”)**.

2.6. Quick Start for pytorch

2.6.1. Without aiDAPTIVLink

*** Regular training procedure ***

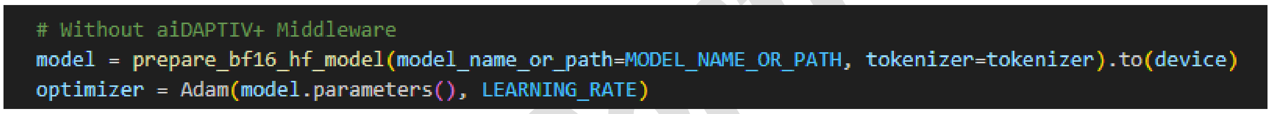
- Import optimizer



```python
# Without aiDAPTIV+ Middleware
from torch.optim import Adam
```

Figure 2-33 Import optimizer

- Create model instance and add model.parameters to optimizer



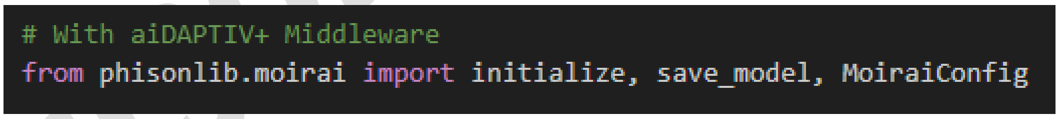
```python
# Without aiDAPTIV+ Middleware
model = prepare_bf16_hf_model(model_name_or_path=MODEL_NAME_OR_PATH, tokenizer=tokenizer).to(device)
optimizer = Adam(model.parameters(), LEARNING_RATE)
```

Figure 2-34 Import optimizer

2.6.2. With aiDAPTIVLink

*** Only Few steps to adapt with aiDAPTIV+***

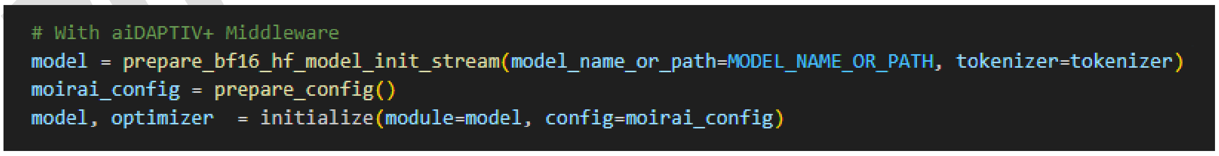
- Import API from site-packages



```python
# With aiDAPTIV+ Middleware
from phisonlib.moirai import initialize, save_model, MoiraiConfig
```

Figure 2-35 Import library

- Create model instance and call initialize. And we’re done!



```python
# With aiDAPTIV+ Middleware
model = prepare_bf16_hf_model_init_stream(model_name_or_path=MODEL_NAME_OR_PATH, tokenizer=tokenizer)
moirai_config = prepare_config()
model, optimizer  = initialize(module=model, config=moirai_config)
```

Figure 2-36 Create model instance and call initialize

3. PERFORMANCE RESULT

In this section, you will learn how to use aiDAPTIV Toolkit to check system performance.

If your hardware configuration is different from the Phison AVL then you should run the aiDPATIV Toolkit benchmark program to test the max batch size for your particular system.

You can refer to section 3.3 to setup your batch size.

3.1. aiDAPTIV Toolkit Installation flow

3.1.1. Download aiDAPTIV Toolkit

```
wget https://phisonbucket.s3.ap-northeast-1.amazonaws.com/aiDAPTIV_Toolkit_2.3.0.zip
```

3.1.2. Unzip download file

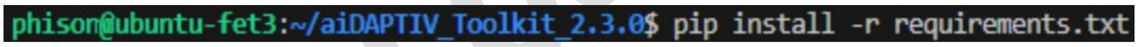
```
unzip aiDAPTIV_Toolkit_2.3.0.zip
```

3.1.3. Change to folder

```
cd aiDAPTIV_Toolkit_2.3.0
```

3.1.4. Deploy aiDAPTIV Toolkit and related library

```
pip install -r requirements.txt
```



```
phison@ubuntu-fet3:~/aiDAPTIV_Toolkit_2.3.0$ pip install -r requirements.txt
```

Figure 3-1 aiDAPTIV Toolkit and related library

3.2. Start aiDAPTIV Toolkit

3.2.1. Modify training setting in aiDAPTIV_Toolkit_2.3.0/project.ini

Default settings can be found in the project.ini file.

```
[ENV_setting]
# Specify the training GPU index
specify_gpu_index =0,1
# GPU number of model training
num_gpus=2
# Training model path in local
model_name_or_path=/home/$USER/Desktop/llm/Llama-3.1-8B-Instruct
# Path of aiDAPTIVCache
nvme_path=/mnt/nvme0
# Password of root
pwd=test

[Performance_test]
# Start batch size of performance test
start_bs=8
# End batch size of performance test
end_bs=100
# Sequence length while training LLM Model
seq_len=2048
# Expect training time in hour
training_hour=0.5
# Enable Triton
triton=True
```

3.2.2. Enter aiDAPTIV Toolkit folder

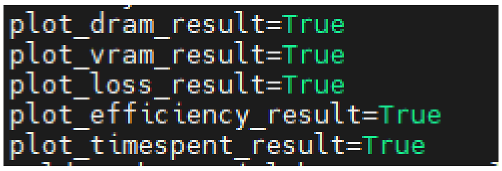
```
cd aiDAPTIV_Toolkit_2.3.0/Script/Model_Test
```

3.2.3. Run aiDAPTIV Toolkit Performance Test

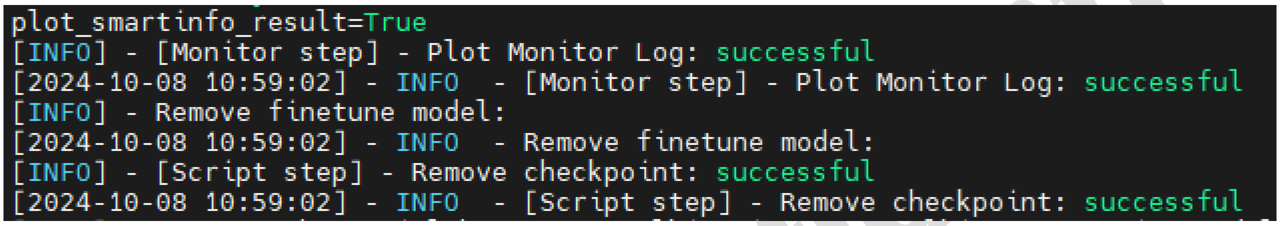
```
python3 aidaptest_run.py --t 1
```

```
 * Type == 1 (--t 1): Test performance
 * Type == 2 (--t 2): Find max batchsize
 * Type == 3 (--t 3): Find max batchsize + Test performance
```

aiDAPTIVTest will stop when the benchmark has finished.



```
plot_dram_result=True
plot_vram_result=True
plot_loss_result=True
plot_efficiency_result=True
plot_timespent_result=True
```



```
plot_smartinfo_result=True
[INFO] - [Monitor step] - Plot Monitor Log: successful
[2024-10-08 10:59:02] - INFO  - [Monitor step] - Plot Monitor Log: successful
[INFO] - Remove finetune model:
[2024-10-08 10:59:02] - INFO  - Remove finetune model:
[INFO] - [Script step] - Remove checkpoint: successful
[2024-10-08 10:59:02] - INFO  - [Script step] - Remove checkpoint: successful
```

Figure 3-2 aiDAPTIV Toolkit Performance Test

- Result

After using the aiDAPTIV Toolkit, the results will be stored in aiDAPTIV_Toolkit_2.3.0/Log directory.

3.3. Performance Result

3.3.1. Log directory Description

1. Figure_4GPU_41bs (Model: Llama-3.1-8B-Instruct)

   This folder will store figure of DRAM usage, GPU usage, Forward time, Backward time, Update time, training Loss and training speed.

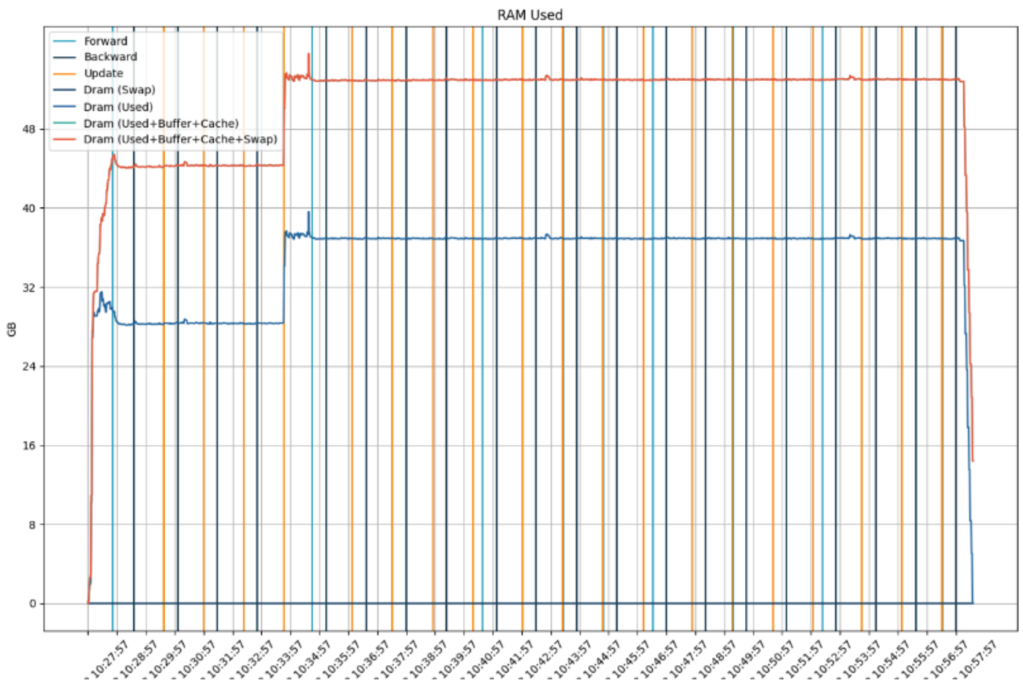
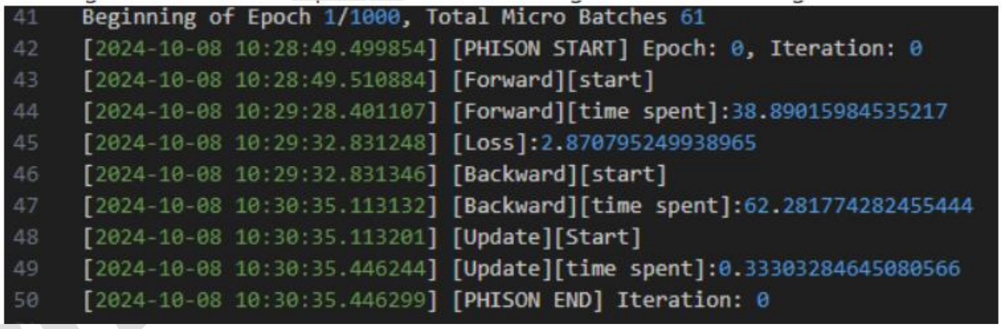


Figure 3-3 Ram Used

2. Training log 4GPU_41bs (Model: Llama-3.1-8B-Instruct)

   This log will store the output of terminal during aiDAPTIV training

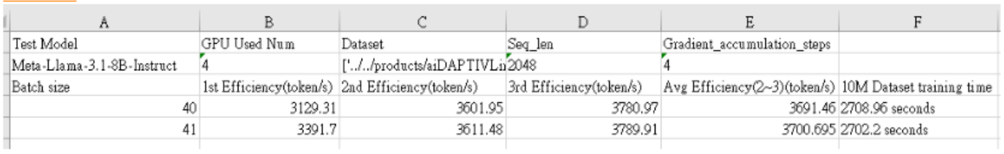


```
Beginning of Epoch 1/1000, Total Micro Batches 61
[2024-10-08 10:28:49.499854] [PHISON START] Epoch: 0, Iteration: 0
[2024-10-08 10:28:49.510884] [Forward][start]
[2024-10-08 10:29:28.401107] [Forward][time spent]:38.89015984535217
[2024-10-08 10:29:32.831248] [Loss]:2.870795249938965
[2024-10-08 10:29:32.831346] [Backward][start]
[2024-10-08 10:30:35.113132] [Backward][time spent]:62.281774282455444
[2024-10-08 10:30:35.113201] [Update][Start]
[2024-10-08 10:30:35.446244] [Update][time spent]:0.33303284645080566
[2024-10-08 10:30:35.446299] [PHISON END] Iteration: 0
```

Figure 3-4 Training Log

3. performance_result.xlsx (Model: Llama-3.1-8B-Instruct)

   This Excel will organize the training information for various parameters in the aiDAPTIV Toolkit Performance test.

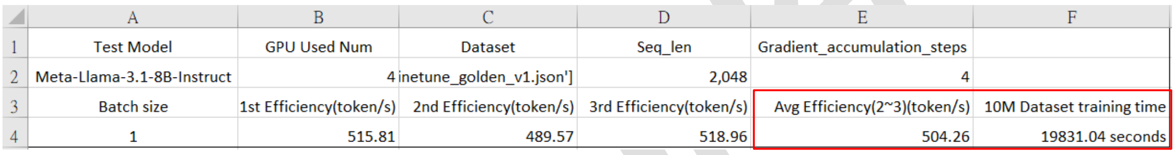


A	B	C	D	E	F
Test Model	GPU Used Num	Dataset	Seq_len	Gradient_accumulation_steps	
Meta-Llama-3.1-8B-Instruct	4	['../../products/aiDAPTIVLi	2048	4	
Batch size	1st Efficiency(token/s)	2nd Efficiency(token/s)	3rd Efficiency(token/s)	Avg Efficiency(2~3)(token/s)	10M Dataset training time
40	3129.31	3601.95	3780.97	3691.46	2708.96 seconds
41	3391.7	3611.48	3789.91	3700.695	2702.2 seconds

Figure 3-5 Example of Performance Result

3.3.2. *Performance Reference*

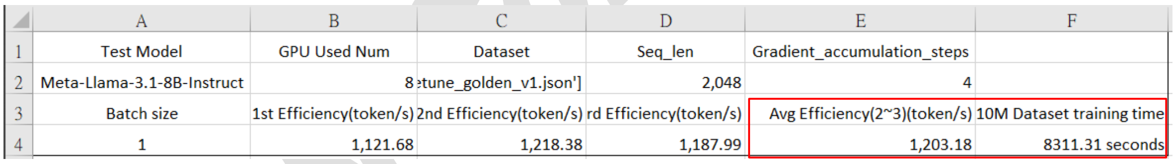
- The maximum batch size’s average efficiency and the time required to train a 10M dataset can be found in the Performance Info sheet of performance_result.xlsx (see the red box in the figure below).
- Following data just for reference. The data would affect by other device (ex: CPU/RAM…etc).
- The data sequence length = 2048 (with triton)
- Model: Llama-3.1-8B-Instruct (with triton)
  - A4000ada



	A	B	C	D	E	F
1	Test Model	GPU Used Num	Dataset	Seq_len	Gradient_accumulation_steps	
2	Meta-Llama-3.1-8B-Instruct	4	inetune_golden_v1.json']	2,048	4	
3	Batch size	1st Efficiency(token/s)	2nd Efficiency(token/s)	3rd Efficiency(token/s)	Avg Efficiency(2~3)(token/s)	10M Dataset training time
4	1	515.81	489.57	518.96	504.26	19831.04 seconds

Figure 3-6 Llama-3.1-8B-Instruct’s Performance Result on 4000ada*4

  - A6000



	A	B	C	D	E	F
1	Test Model	GPU Used Num	Dataset	Seq_len	Gradient_accumulation_steps	
2	Meta-Llama-3.1-8B-Instruct	8	ːtune_golden_v1.json']	2,048	4	
3	Batch size	1st Efficiency(token/s)	2nd Efficiency(token/s)	rd Efficiency(token/s)	Avg Efficiency(2~3)(token/s)	10M Dataset training time
4	1	1,121.68	1,218.38	1,187.99	1,203.18	8311.31 seconds

Figure 3-7 Llama-3.1-8B-Instruct’s Performance Result on A6000*8

- You can optimize hardware configuration to enhance training efficiency and reduce training dataset time.

4. DATASET CONFIGURATION

4.1. QA datasets

All QA datasets (local or from Huggingface) should be configured in “QA_dataset_config.yaml\”.

- **Rules**
  - Use **- - -** to seperate multiple dataset
  - Follow below format, where **DATASET NAME** must be different.

```
<DATASET NAME 1>:
  data_path:
  strategy: "qa"
  system_prompt:
  user_prompt:
  question_key:
  answer_key:
  exp_type:
  label_key:
---
<DATASET NAME 2>:
  .
  .
  .
```

- Description for each value

Key	Description
data_path	local path or huggingface dataset repository name
strategy	**qa**
system_prompt	system_prompt
user_prompt	user prompt should include **{question}** for question insertion (ex. user_prompt : solve the question {question})
question_key	key for question **(NOT SUPPORT NESTED FORMAT)**
answer_key	key for answer **(NOT SUPPORT NESTED FORMAT)**
exp_type	**train** or **inference** or **eval**
label_key	same as answer key

NOTE

If dataset is multiturn chat, the dataset should be converted to **question_key : [question1, question2, question3, …], answer_key : [answer1, answer2, answer3, …]**

4.2. Pretrain datasets

All Pretrain datasets (local or from Huggingface) should be configured in Pretrain_dataset_config.yaml

- **Rules**
  - Use - - - to seperate multiple dataset
  - Follow following format, where **DATASET NAME** must be different

```
<DATASET NAME>:
  data_path:
  strategy: "pretrain"
  system_prompt:
  user_prompt:
  text:
  exp_type:  ## train or inference or eval
---
<DATASET NAME>:
  .
  .
  .
```

- Description for each value

Key	Description
data_path	local path or huggingface dataset repository name
strategy	**pretrain**
system_prompt	system_prompt
user_prompt	user prompt should include **{question}** for question insertion (ex. user_prompt : solve the question {question})
text	key name for pretrain text **(NOT SUPPORT NESTED FORMAT)**
exp_type	**train** or **inference** or **eval**

4.3. RAG datasets

All RAG datasets (local or from Huggingface) should be configured in “RAG_dataset_config.yaml”.

- **Rules**
  - Use **- - -** to seperate multiple dataset
  - follow the format, where **DATASET NAME** must be different

```
<DATASET NAME>:
  data_path:
  strategy: "rag"
  system_prompt:
  user_prompt:
  question_key:
  answer_key:
  rag_key:
  exp_type:
  label_key:
---
<DATASET NAME>:
  .
  .
  .
```

4.4. Description for each value

Key	Description
data_path	local path or huggingface dataset repository name
strategy	**rag**
system_prompt	[optional] system_prompt
user_prompt	user prompt should include **{question}** for question insertion and **{rag}** for rag data insertion (ex. Context: {rag}, based on the context answer the question : {question})
question_key	key for question
answer_key	key for answer
rag_key	key for rag data, **(SUPPORT NESTED FORMAT)** **
exp_type	**train** or **inference** or **eval**
label_key	same as answer key

```
** if RAG data has format
{context : {sentences : [RAG DATA]}}
the rag_key can be given


rag_key:
 - context
 - sentences
```

NOTE
if dataset is multiturn chat, the dataset should be convert to **question_key : [question1, question2, question3, …], answer_key : [answer1, answer2, answer3, …]**

4.4.1. Example 1 (QA dataset)

```
{
    "question": "......",
    "cot_answer": "........"
},
<DATASET NAME>:
  data_path:
  strategy: "qa"
  system_prompt:
  user_prompt: answer the following question {question}
  question_key: question
  answer_key: cot_answer
  exp_type:
  label_key:
```

4.4.2. Example 2 (RAG dataset)

```
{
    "question": [
        "....",
        "....",
        "....."
    ],
   "answer": [
        "....",
        "....",
        "....."
    ],
    "hybrid_chunks": [
        "....",
        "....",
        "....",
    ]
},
```

```
<DATASET NAME>:
  data_path:
  strategy: "rag"
  system_prompt:
  user_prompt: you are a doctor with this background knowledge {rag}, now solve the
following question {question}
```

```
  question_key: question
  answer_key: answer
  rag_key:
   - hybrid_chunks
  exp_type:
  label_key: answer
```

4.4.3. Example 3 (nested RAG dataset)

```
"question": "...",
"cot_answer": "...",
"context": {
    "sentences": [
        "....."
    ]
}
```

```
<DATASET NAME>:
  data_path:
  strategy: "rag"
  system_prompt:
  user_prompt: you are a doctor with this background knowledge {rag}, now solve the
following question {question}
  question_key: question
  answer_key: answer
  rag_key:
    - context
    - sentence
    - 0
  exp_type:
  label_key: answer
```

4.5. Compare Change

Original method seen rm-static as pre-train data, hence no system and user prompt and template.

4.5.1. Previous dataloader on Dahoas/rm-static after apply template

```
Human: How can I cook delicata squash?


Assistant: Try cutting it in half, cutting away the seeds and stringy parts, and
laying it flat in a 400°F oven for 1 hour.  You can drizzle it with olive oil and salt
before you roast it if you want.


Human: Can you fry it?


Assistant: Try taking the squash halves you’ve roasted, and frying them in oil at a
medium heat until it begins to brown and crisp on all sides.  Taste it to see if it
needs salt or spices.<|eot_id|>
```

4.5.2. Current dataloader on Dahoas/rm-static after apply template

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>


This is a chat between a user and an artificial intelligence assistant. The assistant
gives helpful, detailed, and polite answers to the user's questions based on the
context. The assistant should also indicate when the answer cannot be found in the
context.<|eot_id|><|start_header_id|>user<|end_header_id|>


solve the question below :


Human: How can I cook delicata squash?


Assistant: Try cutting it in half, cutting away the seeds and stringy parts, and
laying it flat in a 400°F oven for 1 hour.  You can drizzle it with olive oil and salt
before you roast it if you want.


Human: Can you fry it?


Assistant:<|eot_id|><|start_header_id|>assistant<|end_header_id|>


Try taking the squash halves you’ve roasted, and frying them in oil at a medium heat
until it begins to brown and crisp on all sides.  Taste it to see if it needs salt or
spices.<|eot_id|><|start_header_id|>assistant<|end_header_id|>
```

5. MULTIMODALITY DATASET CONFIGURATION

1. When adding dataset, the yaml file under /commands/image-text-to-text/env_configs need to be configured
2. Currently only support:
   - VQA datasets : QA dataset with vision
3. After configured, the yaml file path should be added to commands/env_config/env_config.yaml ``` yaml = data_path:
   - ./dataset_config/image-text-to-text/VQA_dataset_config.yaml```

- to run the default env_config.yaml, the execute path should locate at aiDAPTIV2/commands

5.1. VQA datasets

All QA datasets (local or from Huggingface) should be configured in image-text-to-text/VQA_dataset_config.yaml

- **Rules**
  - Use - - - to seperate multiple dataset
  - Follow following format, where **DATASET NAME** must be different

```
<DATASET NAME>:
  data_path:
  image_folder : ""
  strategy: "qa"
  system_prompt: ""
  user_prompt: "{question}"
  question_key: ""
  answer_key: ""
  image_key : ""
  label_key: ""
  ---
  <DATASET NAME>:
```

5.2. Description for each value

Key	Description
data_path	local path / local folder or huggingface dataset repository
image_folder	The image folder for the custom dataset will be concatenated with the path within the dataset.

Key	Description
strategy	**qa**
system_prompt	system_prompt
user_prompt	user prompt should include **{question}** for question insertion (ex. user_prompt : solve the question {question})
question_key	key for question
rag_key	key for rag data, **(SUPPORT NESTED FORMAT)** **
answer_key	key for answer
image_key	The key for the image content can be a URL, local file path, or a PIL Image. Leave this field blank for text-only datasets.
label_key	same as answer key

5.3. Supported dataset

1. Repository from huggingface: With given repository name we support automatically download from huggingface. (ex. Multimodal-Fatima/OK-VQA_train)

2. Local huggingface dataset: If the dataset is cloned or downloaded from HuggingFace and comprises *.parquet files, designate the root folder as the data_path for subsequent steps.

    - Ex. for following Directory structure, the data_path should be Desktop/OK-VQA_train

```
Desktop/OK-VQA_train
     |- data
          |- train-0000-of-0004.parquet
          |- train-0001-of-0004.parquet
          |-                .
          |-                .
     |-README.md
```

*5.3.1. Custom dataset: The data path should be in the format of ./.json or ./.csv** to properly parse the data*

- Dataset_config example (QA dataset)
  For example, utilize AI4Math/MathVista in HuggingFace for demonstration. The header include [pid, question, image, decode_image, choice, unit, precision, answer, …]

The Dataset configuration can be set up as follows without the need to download the entire dataset locally.

5.3.2. *Example (MathVista QA dataset)*

```
<DATASET NAME>:
    data_path: "AI4Math/MathVista"
    image_folder : ""
    strategy: "qa"
    system_prompt: "you are a helpful asssistant"
    user_prompt: "Please address the following question using the accompanying image.
    {question}"
    question_key: "question"
    answer_key: "answer"
    image_key : "decode_image"
    label_key:  ""
```

APPENDIX A. HOW TO EVALUATE TRAINED MODEL

In this section, you will learn how to evaluate your trained model and how to create your customized eval function .

A.1 Quick Start

Using the following command, you can complete your evaluation task.

```
phisonai2 --env_config <env_config.yaml path> --exp_config <exp_config.yaml path>
```

- Example of exp_config.yaml

```
process_settings:
  num_gpus: 2
  specify_gpus: null
  master_port: 8299
  multi_node_settings:
    enable: False
    master_addr: "127.0.0.1"

run_settings:
  task_type: "text-generation"
  task_mode: "eval" # or "/home/$USER/aiDAPTIV2/text-generation/eval.py"
  per_device_train_batch_size: 1
  per_update_total_batch_size: 4
  num_train_epochs: 1
  max_iter: -1
  max_seq_len: 2048
  triton: True
  weight_file_format: null
  from_config: false
  precision_mode: 1
  enable_save_optimizer_state: false

  model_saver:
    max_num_of_saved_model_on_epoch_end: -1 # If the value is -1, it is equal to
num_train_epochs.
    enable_save_model_on_iteration: false
    max_num_of_saved_model_on_iteration: 2
    num_of_iteration_to_save_model: 2

  lr_scheduler:
    mode: 1
    learning_rate: 0.000007

  optimizer:
    beta1: 0.9
    beta2: 0.95
    eps: 0.00000001
    weight_decay: 0.01

  early_stop:
    enable: false
    min_delta: 0.01
    patience: 2
    verbose: false

  lora:
    enable_lora: false
    lora_rank: 8
    lora_alpha: 16
    lora_task_type: "CAUSAL_LM"
    lora_target_modules: null
```

- Assign eval or eval.py to the task_mode.
- lower perplexity score indicates that the language model can better predict the next word in the sequence.

A.2 Prepare Your Evaluation Data

You can refer to chapter A to create your own evaluation dataset.

A.3 Prepare Your Evaluation Function (Advance)

You can design your eval function in …/aiDAPTIV2/xxx/eval.py. We provide automatic-speech-recognition, fill-mask, and text-generation example in eval.py, you can modify it for customization.

```
--aiDAPTIV2
    | --automatic-speech-recognition
        |-eval.py
    | --fill-mask
        |--eval.py
    | --text-generation
        |--eval.py
```

Evaluation command:

```
phisonai2 --env_config env_config.yaml --exp_config exp_config.yaml
```

- env_config.yaml

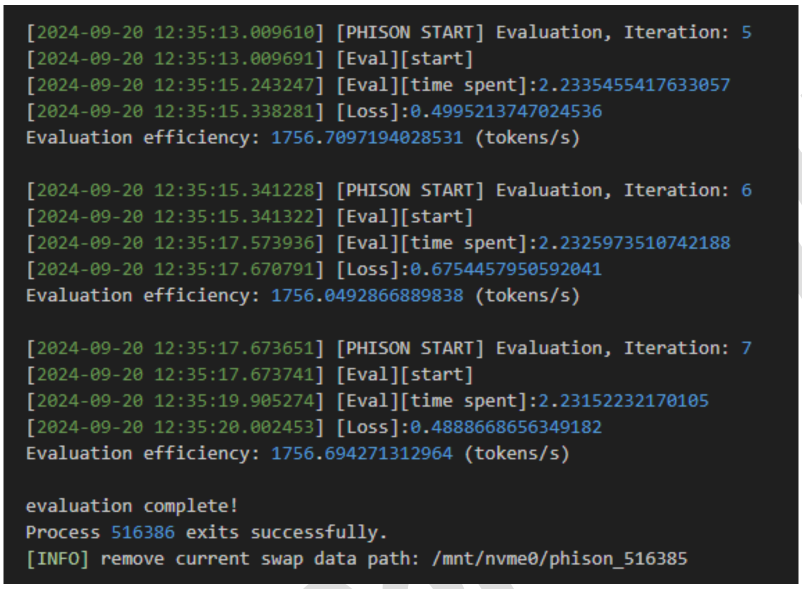
```
path_settings:
  lora:
    lora_weight: ""       # whether to load lora_weight (only activated when lora: true)
    lora_optimizer: ""       # whether to load lora_optimizer (only activated when lora: true)
    lora_output_dir: ""   # whether to save lora adapter weight (only activated when lora: true)
  model_name_or_path: "/home/$USER/output/finetuned_model_2024-09-20-10-36-37/epoch_3_step_7_Llama-3.1-8B-Instruct/"
  multi_node_env_path: null
  optimizer_path: "" # whether to load optimizer
  train_data_path:
    # - ./dataset_config/automatic-speech-recognition/dataset_config.yaml
    # - ./dataset_config/image-text-to-text/VQA_dataset_config.yaml
    # - ./dataset_config/text-generation/Pretrain_dataset_config.yaml
    # - ./dataset_config/text-generation/RAG_dataset_config.yaml
   - ./dataset_config/text-generation/QA_dataset_config.yaml
    # - ./dataset_config/automatic-speech-recognition/dataset_config.yaml
    # - ./dataset_config/image-text-to-text/VQA_dataset_config.yaml
    # - ./dataset_config/text-generation/Pretrain_dataset_config.yaml
    # - ./dataset_config/text-generation/RAG_dataset_config.yaml
    # - ./dataset_config/text-generation/QA_dataset_config.yaml
  nvme_path: "/mnt/nvme0"
  output_dir: ""
  log_name: "Llama-3.1-8B-Instruct_eval.log"
```

- exp_config.yaml

```
process_settings:
  num_gpus: 2
  specify_gpus: null
  master_port: 8299
  multi_node_settings:
    enable: False
    master_addr: "127.0.0.1"

run_settings:
  task_type: "text-generation"
  task_mode: "eval" # or "/home/$USER/aiDAPTIV2/text-generation/eval.py"
  per_device_train_batch_size: 1
  per_update_total_batch_size: 4
  num_train_epochs: 1
  max_iter: -1
  max_seq_len: 2048
  triton: True
  weight_file_format: null
  from_config: false
  precision_mode: 1
  enable_save_optimizer_state: false

  model_saver:
    max_num_of_saved_model_on_epoch_end: -1 # If the value is -1, it is equal to
num_train_epochs.
    enable_save_model_on_iteration: false
    max_num_of_saved_model_on_iteration: 2
    num_of_iteration_to_save_model: 2

  lr_scheduler:
    mode: 1
    learning_rate: 0.000007

  optimizer:
    beta1: 0.9
    beta2: 0.95
    eps: 0.00000001
    weight_decay: 0.01

  early_stop:
    enable: false
    min_delta: 0.01
    patience: 2
    verbose: false
```

```
lora:
  enable_lora: false
  lora_rank: 8
  lora_alpha: 16
  lora_task_type: "CAUSAL_LM"
  lora_target_modules: null
```

- Log file:



```
[2024-09-20 12:35:13.009610] [PHISON START] Evaluation, Iteration: 5
[2024-09-20 12:35:13.009691] [Eval][start]
[2024-09-20 12:35:15.243247] [Eval][time spent]:2.2335455417633057
[2024-09-20 12:35:15.338281] [Loss]:0.4995213747024536
Evaluation efficiency: 1756.7097194028531 (tokens/s)

[2024-09-20 12:35:15.341228] [PHISON START] Evaluation, Iteration: 6
[2024-09-20 12:35:15.341322] [Eval][start]
[2024-09-20 12:35:17.573936] [Eval][time spent]:2.2325973510742188
[2024-09-20 12:35:17.670791] [Loss]:0.6754457950592041
Evaluation efficiency: 1756.0492866889838 (tokens/s)

[2024-09-20 12:35:17.673651] [PHISON START] Evaluation, Iteration: 7
[2024-09-20 12:35:17.673741] [Eval][start]
[2024-09-20 12:35:19.905274] [Eval][time spent]:2.23152232170105
[2024-09-20 12:35:20.002453] [Loss]:0.4888668656349182
Evaluation efficiency: 1756.694271312964 (tokens/s)

evaluation complete!
Process 516386 exits successfully.
[INFO] remove current swap data path: /mnt/nvme0/phison_516385
```

Figure A-1 Log file

APPENDIX B. HOW TO TRAIN MODEL IN DOCKER

In this section, you will learn how to use aiDAPTIVCache and your GPU resources in a docker container.

B.1 Docker Installation option

- Docker official website

https://docs.docker.com/engine/install/ubuntu/

- Download NVIDIA Container Toolkit

https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/latest/install-guide.html

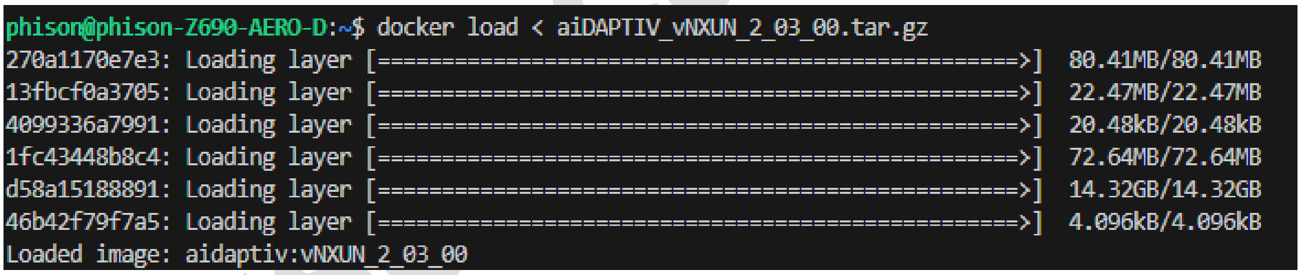
```
# Restart docker service to adopt the change
sudo systemctl restart docker
```

- Docker image

```
wget https://phisonbucket.s3.ap-northeast-1.amazonaws.com/aiDAPTIV_vNXUN_2_03_00.tar.gz
```

- Load docker image

```
docker load < aiDAPTIV_vNXUN_2_03_00.tar.gz
```

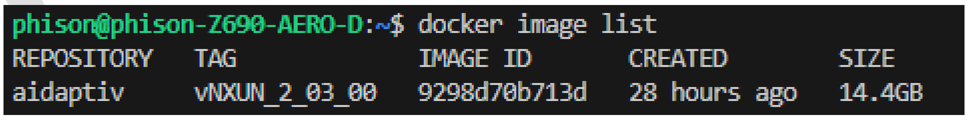


```
phison@phison-Z690-AERO-D:~$ docker load < aiDAPTIV_vNXUN_2_03_00.tar.gz
270a1170e7e3: Loading layer [==================================================>]  80.41MB/80.41MB
13fbcf0a3705: Loading layer [==================================================>]  22.47MB/22.47MB
4099336a7991: Loading layer [==================================================>]  20.48kB/20.48kB
1fc43448b8c4: Loading layer [==================================================>]  72.64MB/72.64MB
d58a15188891: Loading layer [==================================================>]  14.32GB/14.32GB
46b42f79f7a5: Loading layer [==================================================>]  4.096kB/4.096kB
Loaded image: aidaptiv:vNXUN_2_03_00
```

Figure B-1 docker image

- Check docker image list

```
docker image list
```



```
phison@phison-Z690-AERO-D:~$ docker image list
REPOSITORY   TAG            IMAGE ID       CREATED        SIZE
aidaptiv     vNXUN_2_03_00  9298d70b713d   28 hours ago   14.4GB
```

Figure B-2 docker image list

- The command and configuration files needed for the docker deployment will be located in the following folder.

```
/home/root/aiDAPTIV2/commands
```

These files can be modified to match your training project parameters.

```
--commands
    | --env_config
        |--env_config.yaml
    | --exp_config
        |--env_config.yaml
    | --example.sh
```

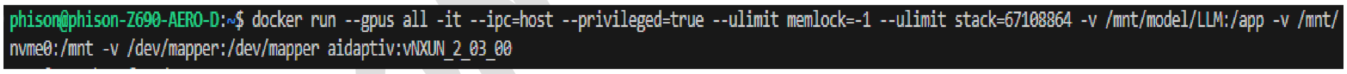
B.2 Run aiDAPTIV Image

If you are deploying the environment using Docker, please execute the following command.   This can be ignored if you are using a native environment.

- docker and nvidia gpu runtime Installation
  - https://docs.docker.com/engine/install/
  - https://docs.nvidia.com/datacenter/cloud-native/container-toolkit/latest/install-guide.html

```
docker run --gpus all -it --ipc=host --privileged=true --ulimit memlock=-1 \
--ulimit stack=67108864 -v </path/to/model>:/app -v </path/to/LVM>:/mnt \
-v /dev/mapper:/dev/mapper -v /var/lock:/var/lock aidaptiv:vNXUN_2_03_00
```

- Successful example



```
phison@phison-Z690-AERO-D:~$ docker run --gpus all -it --ipc=host --privileged=true --ulimit memlock=-1 --ulimit stack=67108864 -v /mnt/model/LLM:/app -v /mnt/
nvme0:/mnt -v /dev/mapper:/dev/mapper aidaptiv:vNXUN_2_03_00
```

Figure B-3 docker successful example

APPENDIX C. HOW TO SET SWAP FILE

- Enable swapping provides extra memory for DRAM. This can extend the range of batch size that you can use if you still have enough memory on the GPU.

```
# Create swap file
sudo dd if=/dev/zero of=/mnt/nvme0/swapfile bs=1M count=256k
# Modify permission
sudo chmod 0600 /mnt/nvme0/swapfile
# Initialize swap file
sudo mkswap /mnt/nvme0/swapfile
# Enable the swap
sudo swapon /mnt/nvme0/swapfile
# Make the swap permanent
sudo echo '/mnt/nvme0/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
```

- If you would like to remove the swap or unplug aiDAPTIVCache, please make sure to follow the steps below to prevent unexpected system issues.

```
# Disable the swap
sudo swapoff /mnt/nvme0/swapfile
# Remove permanent swap setting
sudo sed -i '/\/mnt\/nvme0\/swapfile/d' /etc/fstab
# Remove swapfile (optional)
sudo rm /mnt/nvme0/swapfile
```

APPENDIX D. APPROVED VENDOR LIST (AVL)

D.1 GPU AVL

Table E-1 GPU AVL

Vendor	Product Name	Bus	Memory
NVIDIA	H200	PCIe 4.0 x16	80 GB, HBM2e, 5120 bit
NVIDIA	H100	PCIe 4.0 x16	80 GB, HBM2e, 5120 bit
NVIDIA	RTX A6000	PCIe 4.0 x16	48 GB, GDDR6, 384 bit
NVIDIA	RTX A5000	PCIe 4.0 x16	24 GB, GDDR6, 384 bit
NVIDIA	GeForce RTX 4090	PCIe 4.0 x16	24 GB, GDDR6X, 384 bit
NVIDIA	L40	PCIe 4.0 x16	48 GB, GDDR6, 384 bit
NVIDIA	L40S	PCIe 4.0 x16	48 GB, GDDR6, 384 bit
NVIDIA	RTX 6000 Ada Generation	PCIe 4.0 x16	48 GB, GDDR6, 384 bit
NVIDIA	GeForce RTX 4090 D	PCIe 4.0 x16	24 GB, GDDR6X, 384 bit
NVIDIA	RTX 4000 Ada Generation	PCIe 4.0 x16	20 GB, GDDR6, 160 bit
NVIDIA	RTX 4000 SFF Ada Generation	PCIe 4.0 x16	20 GB, GDDR6, 160 bit
NVIDIA	RTX 5000 Ada Generation	PCIe 4.0 x16	32 GB, GDDR6, 256 bit

D.2 CPU AVL

Table E-2 CPU AVL

Brand	Naming	Count	Cores	Clk	Lanes
Intel	Xeon Gold 5320	2	26	2.2	64
Intel	Xeon Gold 6330	2	28	2	64
Intel	Xeon w5-3425	1	12	3.2	112
Intel	Xeon Gold 6538Y+	2	32	2.2	80
Intel	Xeon Silver 4410T	2	10	2.7	80
Intel	i9-13900	1	24	1.5	20
Intel	i9-12900E	1	16	1.7	20
Intel	Xeon Silver 4410Y	2	8	2	80
Intel	Xeon Silver 5315Y	2	8	3.2	64

AMD	Ryzen Threadripper 7980X 64-Cores	1	64	5.1	92
AMD	EPYC 7713P	1	64	2	128

AMD	EPYC 9174F	1	16	4.1	128

D.3 Support Model list

Table E-3 Support model list

No	Task Type	Model Name
1	Text-Generation	Llama-2-7b-hf
2	Text-Generation	Llama-2-13b-hf
3	Text-Generation	Llama-2-70b-hf
4	Text-Generation	Meta-Llama-3-8B
5	Text-Generation	Meta-Llama-3-70B
6	Text-Generation	Mistral-7B-Instruct-v0.1
7	Text-Generation	Mixtral-8x7B-Instruct-v0.1
8	Text-Generation	Mixtral-8x22B-Instruct-v0.1
9	Text-Generation	CodeLlama-7b-hf
10	Text-Generation	Phind-CodeLlama-34B-v1
11	Text-Generation	CodeLlama-70b-hf
12	Text-Generation	LlamaGuard-7b
13	Text-Generation	falcon-180B
14	Text-Generation	ko-llm-llama-2-7b-LoRA-IA3
15	Text-Generation	b.11.0.0
16	Text-Generation	vicuna-33b-v1.3
17	Text-Generation	Breeze-7B-Instruct-v0_1
18	automatic-speech-recognition	whisper-large-v2
19	Text-Generation	Qwen1.5-0.5B-Chat
20	Text-Generation	Qwen1.5-1.8B-Chat
21	Text-Generation	Qwen1.5-4B-Chat
22	Text-Generation	Qwen1.5-7B-Chat
23	Text-Generation	Qwen1.5-14B-Chat
24	Text-Generation	Qwen1.5-72B-Chat
25	Text-Generation	Qwen1.5-110B-Chat

26	Text-Generation	Yi-1.5-6B
27	Text-Generation	Yi-1.5-9B-Chat
28	Text-Generation	Yi-1.5-34B-Chat
29	Text-Generation	deepseek-llm-7b-chat
30	Text-Generation	deepseek-moe-16b-chat
31	Text-Generation	deepseek-llm-67b-chat
32	Text-Generation	Yuan2-M32-hf
33	Text-Generation	Baichuan-7B
34	Text-Generation	chatglm-6b
35	Text-Generation	Qwen2-72B
36	Fill-Mask	bert-base-uncased
37	Text-Generation	glm-4-9b
38	Text-Generation	Qwen2-7B
39	Text-Generation	Qwen2-72B-Instruct
40	Text-Generation	Baichuan2-7B-Chat
41	Text-Generation	DeepSeek-Coder-V2-Instruct
42	Text-Generation	chatglm3-6b
43	Text-Generation	glm-4-9b-chat
44	Text-Generation	Llama-3.1-8B-Instruct
45	Text-Generation	Meta-Llama-3.1-70b-Instruct
46	Text-Generation	Gemma2-9b-it
47	Text-Generation	Gemma2-27b-it
48	Text-Generation	Llama-3-Taiwan-70B-Instruct
49	Image-Text-to-Text	InternVL2-1B
50	Image-Text-to-Text	InternVL2-2B
51	Image-Text-to-Text	InternVL2-4B
52	Image-Text-to-Text	InternVL2-8B
53	Image-Text-to-Text	InternVL2-26B
54	Image-Text-to-Text	InternVL2-40B
55	Image-Text-to-Text	InternVL2-Llama3-76B

56	Image-Text-to-Text	Llama-3.2-11B-Vision-Instruct
57	Image-Text-to-Text	Llama-3.2-90B-Vision-Instruct
58	Image-Text-to-Text	llava-1.5-7b-hf
59	Image-Text-to-Text	llava-1.5-13b-hf
60	Image-Text-to-Text	Qwen2-VL-2B-Instruct
61	Image-Text-to-Text	Qwen2-VL-7B-Instruct
62	Image-Text-to-Text	Qwen2-VL-72B-Instruct
63	Text-Generation	Qwen2.5-72B-Instruct
64	Image-Text-to-Text	Pixtral-12B
65	Text-Generation	Smaug-72B-v0.1
66	Image-Text-to-Text	chameleon-7b
67	Image-Text-to-Text	chameleon-30b
68	Image-Text-to-Text	Phi-3-vision-128k-instruct
69	Image-Text-to-Text	Phi-3.5-vision-instruct
70	Text-Generation	Llama-3.3-70B-Instruct
71	Text-Generation	DeepSeek-R1-Distill-Llama-70B
72	Text-Generation	DeepSeek-R1-Distill-Qwen-32B
73	Text-Generation	QwQ-32B
74	Text-Generation	LongWriter-glm-9b
75	Text-Generation	Qwen-2-0.5B
76	Text-Generation	Qwen2.5-0.5B-Instruct
77	Text-Generation	Llama-3.2-3B-Instruct
78	Image-Text-to-Text	llava-v1.6-vicuna-7b-hf
79	Text-Generation	Phi-3.5-mini-instruct
80	Text-Generation	grantie-3.0-8b-instruct
81	Text-Generation	gemma-3-1b-it
82	Image-Text-to-Text	gemma-3-27b-it
83	automatic-speech-recognition	whisper-large-v3-turbo
84	Image-Text-to-Text	Qwen2.5-VL-7B-Instruct
85	Image-Text-to-Text	Qwen2.5-VL-72B-Instruct

86	Text-Generation	Phi-4
87	Text-Generation	Phi-4-reasoning
88	Text-Generation	Qwen3-0.6B
89	Text-Generation	Qwen3-32B
90	Text-Generation	Mistral-Small-3.1-24B-Instruct
91	Text-Generation	Deepseek-v3-bf16