



晶創25 (Nano 5)

- 2025/06
- 計算力 13.06 PFLOPS
- Top500 # 118
- Green500 # 72

How to use Nano5 of NCHC?

Shih-Hsin Chen,

Associate Professor and Chairman,

Department of Computer Science and Information Engineering,

Tamkang University

Outline

- Introduction
- Login to Nano5 server
- Train an image classification model
- Slurm file explanations
- Conclusions

1. Introduction

- Nano5: The latest supercomputer of NCHC released in 2025/4.
- Support H100/H200 GPUs, multi-core CPU, RAM, and HFS.
- H100 has the TF32 (989 TFLOPS) and BF16
- Nano5 runs the container of Singularity Image Format (SIF).
- nVIDIA NGC container could be directly employed.

| GPU Spec | VRAM | # of GPUs on Nano5 |
|----------|--------------|--------------------|
| H100 | 80 GB HBM3 | 168 |
| H200 | 141 GB HBM3e | 128 |

1. Introduction

- **Hourly Pricing and performance**

| NCHC Platform | GPU Spec | FP32 | TF32 | FP16/BF16 | NSTC | Academic |
|---------------|------------|-------------|------------|-------------|------|----------|
| Nano5 | H100 80GB | 67 TFLOPS | 989 TFLOPS | 1979 TFLOPS | \$25 | \$50 |
| Nano5 | H200 141GB | 67 TFLOPS | 989 TFLOPS | 1979 TFLOPS | \$30 | \$60 |
| TWCC | V100 32 GB | 15.7 TFLOPS | - | 125 TFLOPS | \$10 | \$20 |

- TF32 is on automatically without revising the code (e.g., loss scaling). Manually control of the TF32 is shown below.
- H100 has a better cost-performance ratio than V100 on TWCC (Taiwan Computing Cloud, or called TAIWANIA 2)

```
import torch
torch.backends.cuda.matmul.allow_tf32 = True
torch.backends.cudnn.allow_tf32 = True
```

1. Introduction

- NCHC uses Slurm (Simple Linux Utility for Resource Management) across the TWCC, TAIWANIA 3, Forerunner 1 (F1), and Nano5.
- Slurm is an open-source, highly scalable workload manager and job scheduler for Linux clusters that handles resource allocation, job queuing/scheduling, and accounting.
- Two important commands (srun and sbatch) is shown in the explanation of the Slurm file.

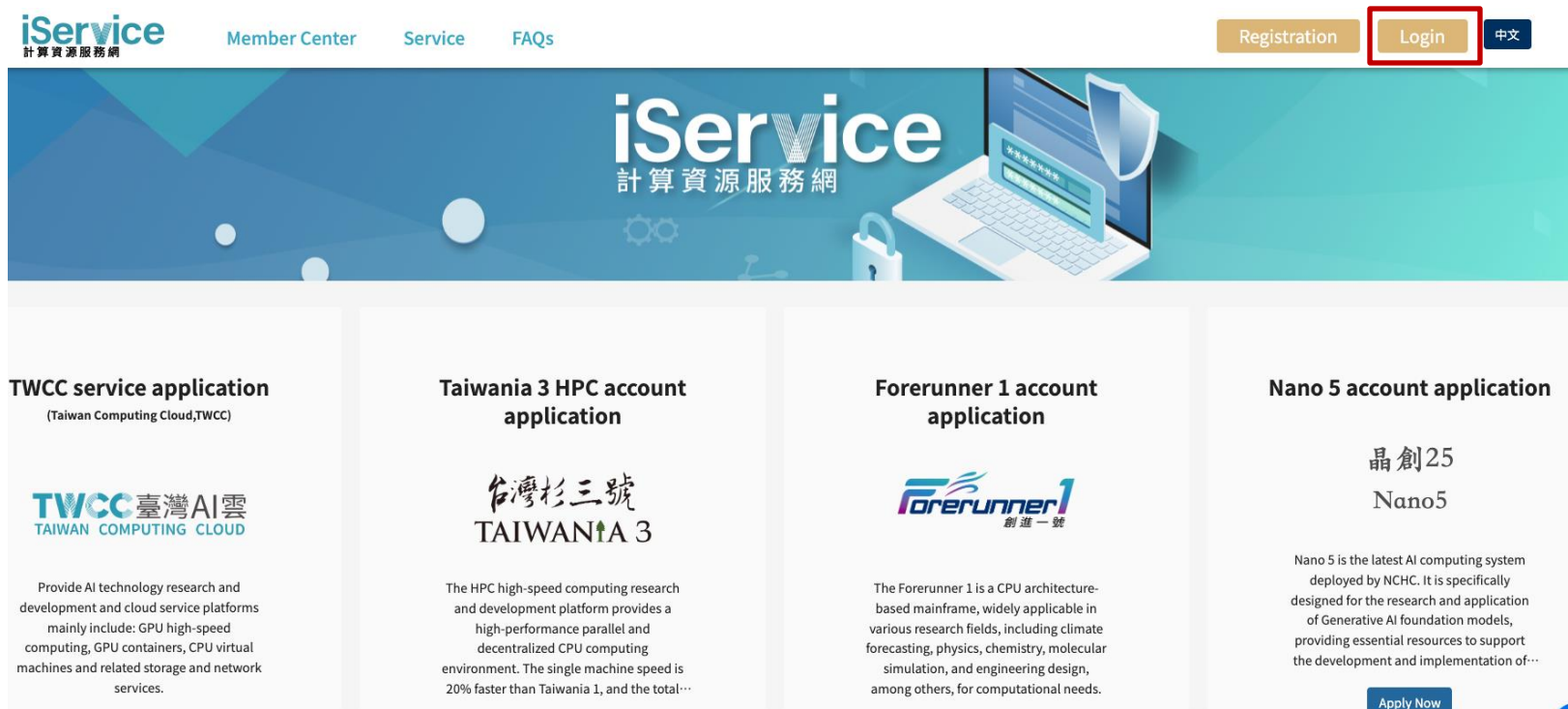
2. Login to Nano5 Server

- Three critical major topics here:
 - Set the server password
 - Connection by SSH client
 - Upload files/directories by FileZilla



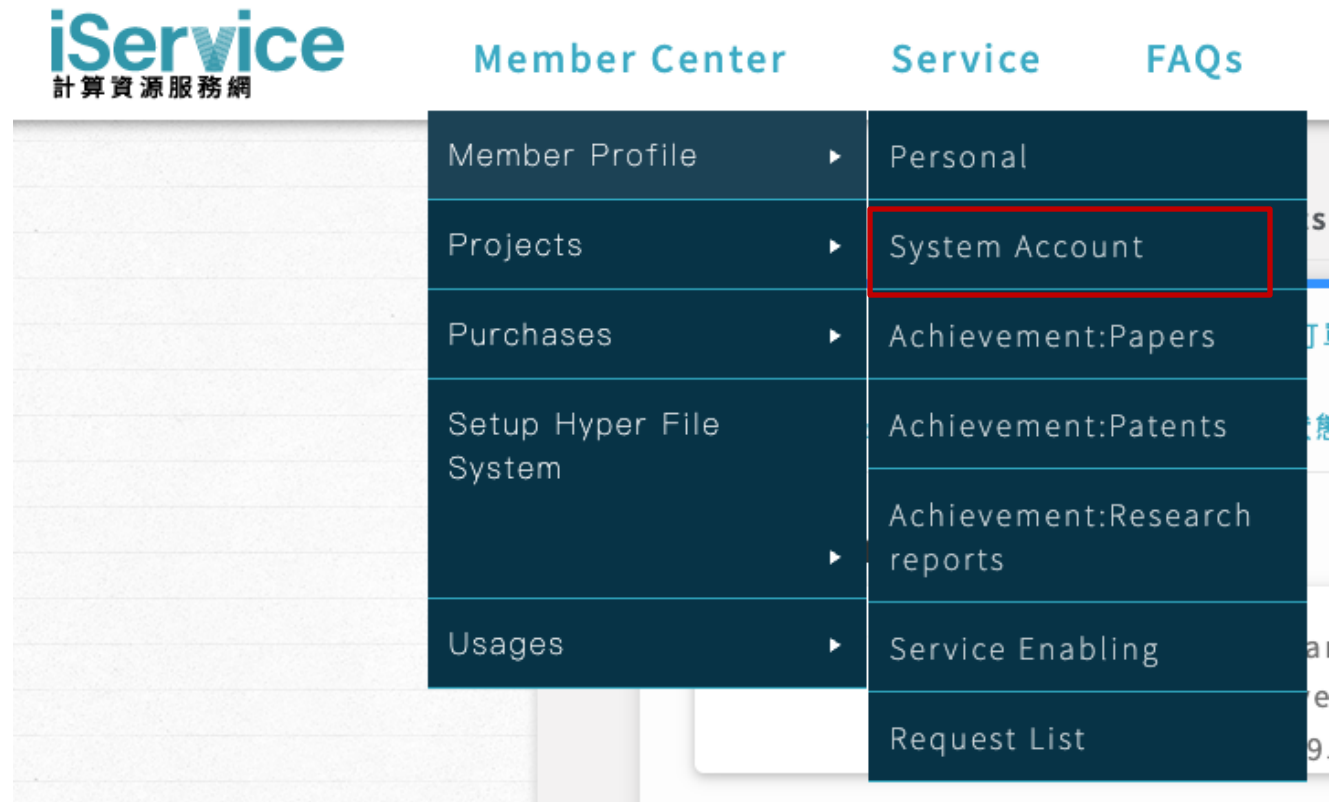
2.1 Set the server password

- Step 1: Login to HCHC Service Portal
- https://iservice.nchc.org.tw/nchc_service/index.php



2.1 Set the server password

- Step 2: Select your project
- Step 3: Select Member Center/Member Profile/System Account



2.1 Set the server password

- Step 4: Record your account name (Orange area below)
- Step 5: Click “Change unix account password”
- Step 6 (Optional): OTP by mobile App for faster 2-factor authentication

Member Center Service FAQs

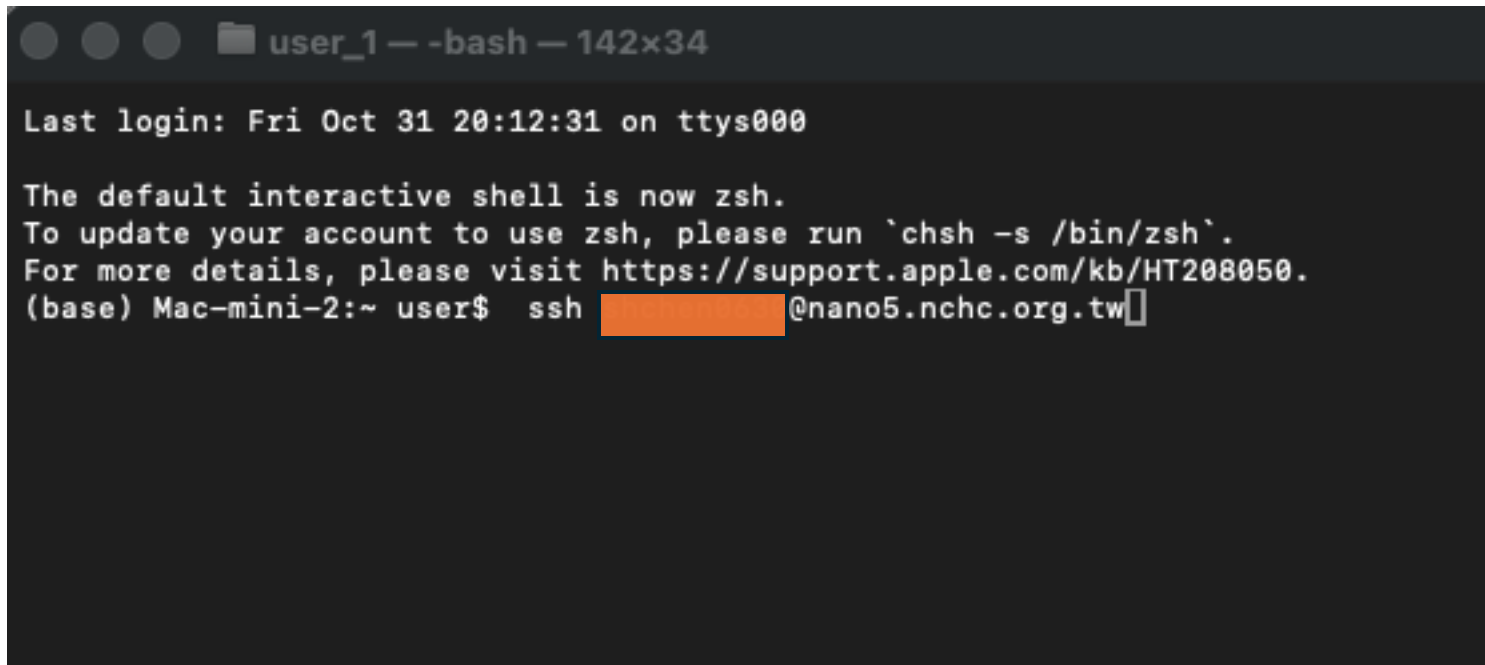
Modify the basic information of the supercomputer account.

Supercomputer account

| | | |
|---|---|---|
| Supercomputer account <div><div></div>(enable)</div> | Supercomputer password <div>change unix account password</div> <div>!!!Date of last supercomputer password change. : 2025-10-31 20:19:53</div> | OTP Token: Valid <div>delete user otp</div> <div>Date of last successful MOTP token deletion : 無</div> |
|---|---|---|

2.2 Connection by SSH client

- Step 1: We use the SSH to connect to the Nano5 Linux server
 - `ssh yourAccount@nano5.nchc.org.tw`
- If you use Windows, please use Putty.

A terminal window titled 'user_1 — -bash — 142x34' showing the output of an SSH connection. The text displayed is: 'Last login: Fri Oct 31 20:12:31 on ttys000', 'The default interactive shell is now zsh.', 'To update your account to use zsh, please run `chsh -s /bin/zsh`.', 'For more details, please visit https://support.apple.com/kb/HT208050.', and '(base) Mac-mini-2:~ user\$ ssh [redacted]@nano5.nchc.org.tw'.

```
user_1 — -bash — 142x34
Last login: Fri Oct 31 20:12:31 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) Mac-mini-2:~ user$ ssh [redacted]@nano5.nchc.org.tw
```

2.2 Connection by SSH client

- Step 2: Select a 2-factor login method. We use 3. Email OTP here.

```
[(base) Mac-mini-2:~ user$ ssh shchen0630@nano5.nchc.org.tw
(shchen0630@nano5.nchc.org.tw) Please select the 2FA login method.
1. Mobile APP OTP
2. Mobile APP PUSH
3. Email OTP
Login method: 3
```

2.2 Connection by SSH client

- Step 3: Enter your password first and then check your email to obtain the OTP code.

[NCHC iService 服務網]登入驗證碼通知信，此密碼將於2025-11-02 1

◆ Summarize this email



國家高速網路與計算中心

to me ▾

您好：

412289 為您的[NCHC iService 服務網] 登入驗證碼，

您於 2025-11-02 19:07:10 使用主機帳號 [redacted] 登入 140.110.148.3，

此密碼將於 2025-11-02 19:10:10 內有效。

如果您未提交驗證請求，敬請儘速向本中心反應。

```
[(base) Mac-mini-2:~ user$ ssh [redacted]@nano5.nchc.org.tw
([redacted]@nano5.nchc.org.tw) Please select the 2FA login method.
1. Mobile APP OTP
2. Mobile APP PUSH
3. Email OTP
Login method: 3
[(redacted]@nano5.nchc.org.tw) Password:
([redacted]@nano5.nchc.org.tw) OTP: 412289
```



2.2 Connection by SSH client

- Step 4: Login successfully!

```
to the bottom of the page
=====
Latest update: 2025-11-02 19:08:02

```

| | #1 | #2 | #3 | #4 | #5 | #6 | #7 | #8 | %CPU | State |
|--------|---------|---------|---------|---------|---------|---------|--------|--------|-------|---------------------------|
| hgpn01 | | | | | | | | | 16.26 | IDLE |
| hgpn02 | <64589> | <64642> | <64797> | <64800> | <64842> | <64843> | | | 28.02 | MIXED |
| hgpn03 | <64487> | <~ | ~ | ~ | ~ | 64609~ | ~ | ~ | 26.74 | MIXED |
| hgpn04 | | | | | | | | | 0.00 | DOWN+DRAIN+NOT_RESPONDING |
| hgpn05 | <64488> | <64509> | | | | | | | 26.10 | MIXED |
| hgpn06 | <~ | ~ | ~ | 64656~ | ~ | ~ | 64658~ | ~ | 26.09 | MIXED |
| hgpn17 | <~ | ~ | ~ | ~ | 64281~ | ~ | ~ | ~ | 22.27 | MIXED |
| hgpn18 | <~ | ~ | ~ | ~ | 64705~ | ~ | ~ | 64730~ | 21.93 | MIXED+COMPLETING |
| hgpn19 | <~ | ~ | ~ | ~ | ~ | 64680~ | ~ | ~ | 24.12 | MIXED |
| hgpn20 | <~ | ~ | ~ | ~ | ~ | 64681~ | ~ | ~ | 24.38 | MIXED |
| hgpn21 | <~ | ~ | ~ | ~ | ~ | 64581~ | ~ | ~ | 53.52 | MIXED |
| hgpn39 | <~ | ~ | ~ | 64555~ | ~ | ~ | 64563~ | ~ | 24.56 | MIXED |
| hgpn40 | <~ | ~ | ~ | ~ | 64825~ | ~ | ~ | ~ | 22.06 | MIXED |
| hgpn41 | <~ | ~ | ~ | ~ | ~ | 64699~ | ~ | ~ | 41.44 | MIXED |
| hgpn42 | <~ | ~ | ~ | ~ | ~ | 64296~ | ~ | ~ | 28.61 | MIXED |
| hgpn43 | <~ | ~ | ~ | ~ | ~ | 64296~ | ~ | ~ | 29.44 | MIXED |
| hgpn44 | <~ | ~ | ~ | ~ | ~ | 64296~ | ~ | ~ | 29.52 | MIXED |
| hgpn45 | <~ | ~ | ~ | ~ | ~ | 64296~ | ~ | ~ | 29.23 | MIXED |
| hgpn46 | <64290> | | | | | | | | 16.22 | MIXED |

```
Load Average: 17.49 17.77 18.46
19.1 u5010945
Sun Nov 2 15:47:09 2025 from 42.70.208.163
@cbi-lgn01 ~]$
```

2.3 Upload files by FileZilla

- Step 1: Download the FileZilla
- Step 2: Create a new site. We use SFTP protocol from the dropdown menu.



2.3 Upload files by FileZilla

- Step 3: Input Nano5 address (nano5.nchc.org.tw), port number 2222, *select the interactive login*, and your account.
- Interactive model will promote the login method, password, and code.



2.3 Upload files by FileZilla

- Step 3: After you click connect, there is a pop-up window. We input 3 (Email OTP) again. You input the password and OTP code.



輸入密碼

請輸入此伺服器的密碼:

名稱: TWCC-Nano5

主機: nano5.nchc.org.tw:2222

使用者:

需要確認:

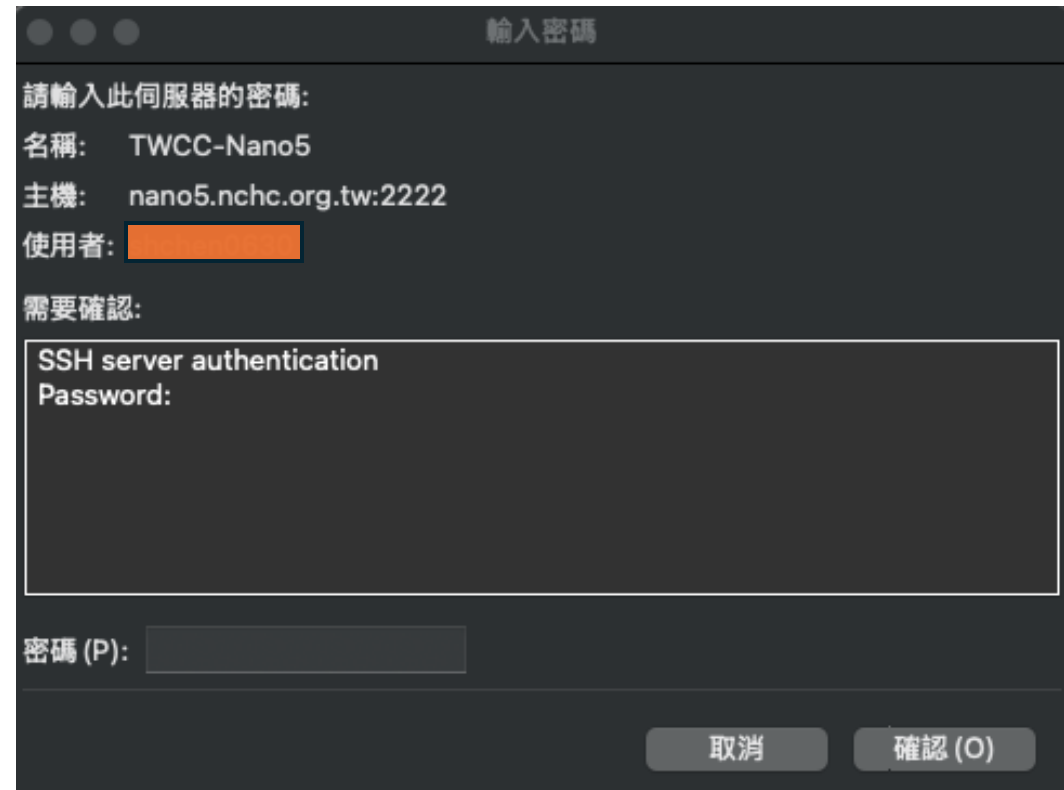
SSH server authentication

Please select the 2FA login method. 1. Mobile APP OTP 2. Mobile APP PUSH

3. Email OTP Login method:

密碼 (P):

取消 確認 (O)



輸入密碼

請輸入此伺服器的密碼:

名稱: TWCC-Nano5

主機: nano5.nchc.org.tw:2222

使用者:

需要確認:

SSH server authentication

Password:

密碼 (P):

取消 確認 (O)

2.3 Upload files by FileZilla

- Once it is connected, you could try to upload a file or a folder.
- The directory is under your HOME on the Nano5 Linux server.
- Extra: Try the Mobile App OTP method
 - After you make a successful connect to SFTP server, it is worthwhile to enable this method.
 - This method works more efficient.

3. Train an image classification model

- We will upload the deep learning program and dataset.
- Besides, we write a Slurm script to deploy the training job(s).
- For simplicity, a demo project is provided.
- Please run git clone command the project on Nano5 :
 - `git clone https://github.com/worldstar/NCHC-Slurm-Demo`

3.1 Download a demo GitHub Project

- Credit of this all-in-one GitHub project
- **A. Image Classification Program:** timm (PyTorch Image Models)
 - <https://github.com/huggingface/pytorch-image-models.git>
 - A library of state-of-the-art PyTorch vision models (ResNet, EfficientNet, ConvNeXt, ViT, Swin Transformer) with pretrained weights plus training, inference, and evaluation utilities.
 - Maintained by Hugging Face now

3.1 Download a demo GitHub Project

- Credit of this all-in-one GitHub project
- **B. Dataset**
 - Diabetic Retinopathy Screening AI Computer Vision Model
 - <https://universe.roboflow.com/ucla-master-of-quantitative-economics/diabetic-retinopathy-screening-ai>

The screenshot displays the Roboflow dataset management interface. At the top, there are two buttons: 'Use this Dataset' and 'Use this Model'. Below these, the 'Versions' section shows a list of dataset versions, with the current version 'v1' (dated 2023-04-19 12:45am) selected and marked with a checkmark. To the right of the version list, a 'Download Dataset' button is highlighted with a red rectangle. Below the version list, the dataset details for 'v1' are shown, including the total number of images (2838) and a 'View All Images' link. A row of eight sample images of retinal fundus is displayed. Below the images, the 'Dataset Split' section shows the distribution: TRAIN SET (70%, 1986 Images), VALID SET (20%, 568 Images), and TEST SET (10%, 284 Images). At the bottom, the 'Preprocessing' section indicates that 'Auto-Orient' is applied and images are 'Resize: Fit within 192x192'.

Use this Dataset Use this Model

Versions

v1 2023-04-19 12:45am
Generated on Apr 19, 2023

Download Dataset

2838 Total Images View All Images →

Dataset Split

| Split | Percentage | Count |
|-----------|------------|-------------|
| TRAIN SET | 70% | 1986 Images |
| VALID SET | 20% | 568 Images |
| TEST SET | 10% | 284 Images |

Preprocessing

Auto-Orient: Applied
Resize: Fit within 192x192

3.2 Revise the Slurm file

- Please edit the singularityClassification.slurm by text editor (vi or vim)
- Modify the account name is your **project ID**, starting with MSTXXXX or ACDXXXX.
- We explain the parameters. But please note the log files are with .out and .err sub filename.
- %x is job-name (cls-sing)
- %j is the job ID

```
#!/bin/bash
#SBATCH --job-name=cls-sing
#SBATCH --partition=dev
#SBATCH --account=MSTXXXX ## iService_ID 計畫 ID
## 選一種 GPU 申請方式（依叢集設定）
# 新版寫法（若支援）
##SBATCH --gpus=1
# 通用寫法（多數中心可用）
#SBATCH --gres=gpu:1
#SBATCH --cpus-per-task=4
#SBATCH --time=01:00:00
#SBATCH --output=%x-%j.out
#SBATCH --error=%x-%j.err
```

3.3 Run the Slurm file

- Step 1: Connect to the Nano5 by SSH.
- Step 2: Enter the project directory
 - `cd ~/NCHC-Slurm-Demo/`
- Step 3: Run the sbatch command
 - `sbatch singularityClassification.slurm`
- Step 4: You will read the job ID from the terminal

3.3 Run the Slurm file

- Step 5: Check your status of your batch job. It might be PD in the beginning
 - `queue --job your_JOB_ID`
- Step 6: Read the output log: `cls-sing-XXXXX.out`

```
NVIDIA-SMI 550.127.08 Driver Version: 550.127.08 CUDA Version: 12.4
+-----+-----+-----+-----+-----+-----+-----+-----+
| GPU | Name | Perf | Persistence-M | Bus-Id | Disp.A | Volatile Uncorr. ECC |
| Fan | Temp |      | Pwr:Usage/Cap |        | Memory-Usage | GPU-Util | Compute M. |
|-----+-----+-----+-----+-----+-----+-----+-----+
| 0   | NVIDIA H100 80GB HBM3 | P0 | On | 00000000:BC:00:0 | Off | 0 |
| N/A | 28C |      | 70W / 700W | 1MiB / 81559MiB | 0% | Default |
|-----+-----+-----+-----+-----+-----+-----+-----+

Processes:
+-----+-----+-----+-----+-----+-----+-----+-----+
| GPU | GI | CI | PID | Type | Process name | GPU Memory |
| ID  | ID | ID |      |      |               | Usage      |
+-----+-----+-----+-----+-----+-----+-----+-----+
| No running processes found |

pip 25.3 from /home, .local/lib/python3.10/site-packages/pip (python 3.10)
Looking in indexes: .org/simple, https://pypi.ngc.nvidia.com
Requirement already satisfied: pip==25.3 in /home/shchen0630/.local/lib/python3.10/site-packages (25.3)
pip 25.3 from /home, .local/lib/python3.10/site-packages/pip (python 3.10)
Looking in indexes: .org/simple, https://pypi.ngc.nvidia.com
Requirement already satisfied: pip in /home/shchen0630/.local/lib/python3.10/site-packages (25.3)
Looking in indexes: https://pypi.org/simple, https://pypi.ngc.nvidia.com
Requirement already satisfied: wheel in /home/shchen0630/.local/lib/python3.10/site-packages (0.45.1)
Requirement already satisfied: setuptools in /home/shchen0630/.local/lib/python3.10/site-packages (80.9.0)
Looking in indexes: https://pypi.org/simple, https://pypi.ngc.nvidia.com
Requirement already satisfied: pip in /home/shchen0630/.local/lib/python3.10/site-packages (25.3)
```

3.4 Access Output Results

- Step 1: Read the error log: cls-sing-XXXXX.err under the same project folder

```
13:4: not a valid test operator: (
13:4: not a valid test operator: 550.127.08
Training with a single process on 1 device (cuda).
Model convnext_base created, param count:88591464
Data processing configuration for current model + dataset:
  input_size: (3, 512, 512)
  interpolation: bicubic
  mean: (0.485, 0.456, 0.406)
  std: (0.229, 0.224, 0.225)
  crop_pct: 0.875
  crop_mode: center
Created AdamW (adamw) optimizer: lr: 0.0005, betas: (0.9, 0.999), eps: 1e-08, weight_decay: 2e-05, amsgrad: False, foreach: None, maxin
False, differentiable: False, fused: None
Using native Torch AMP. Training in mixed precision.
Scheduled epochs: 50 (epochs + cooldown_epochs). Warmup within epochs when warmup_prefix=False. LR stepped per epoch.
Train: 0 [ 0/31 ( 3%)] Loss: 7.10 (7.10) Time: 22.751s, 2.81/s (22.751s, 2.81/s) LR: 1.000e-05 Data: 1.496 (1.496)
Test: [ 0/8] Time: 0.781 (0.781) Loss: 4.566 ( 4.566) Acc@1: 0.000 ( 0.000) Acc@5: 100.000 (100.000)
Test: [ 8/8] Time: 1.299 (0.300) Loss: 7.317 ( 2.339) Acc@1: 0.000 ( 50.528) Acc@5: 0.000 ( 88.732)
Current checkpoints:
('./output/train/20251102-111549-convnext_base-512/checkpoint-0.pth.tar', 50.528169014084504)

Train: 1 [ 0/31 ( 3%)] Loss: 4.15 (4.15) Time: 0.961s, 66.62/s (0.961s, 66.62/s) LR: 1.000e-04 Data: 0.685 (0.685)
Test: [ 0/8] Time: 0.511 (0.511) Loss: 2.922 ( 2.922) Acc@1: 0.000 ( 0.000) Acc@5: 100.000 (100.000)
Test: [ 8/8] Time: 0.078 (0.152) Loss: 3.780 ( 1.539) Acc@1: 0.000 ( 50.704) Acc@5: 96.429 ( 99.648)
Current checkpoints:
('./output/train/20251102-111549-convnext_base-512/checkpoint-1.pth.tar', 50.70422535211268)
('./output/train/20251102-111549-convnext_base-512/checkpoint-0.pth.tar', 50.528169014084504)
```

3.4 Access Output Results

- Step 2: Model Files and Summary.csv. Please go to ~/src/ NCHC-Slurm-Demo/output/train/XXX-convnext_base-512/

| 檔案名稱 ^ | 檔案大小 | 檔案類型 | 最後修改時間 | 權限 |
|-----------------------|---------------|---------|---------------------------|------------|
| .. | | | | |
| args.yaml | 2,870 | yaml-檔案 | 2025/11/02 11時 15 分 49... | -rw-r--r-- |
| checkpoint-33.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 22 分 4... | -rw-r--r-- |
| checkpoint-36.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 23 分 17 秒 | -rw-r--r-- |
| checkpoint-38.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 23 分 4... | -rw-r--r-- |
| checkpoint-39.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 23 分 51... | -rw-r--r-- |
| checkpoint-41.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 24 分 14... | -rw-r--r-- |
| checkpoint-45.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 25 分 0... | -rw-r--r-- |
| checkpoint-46.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 25 分 11 秒 | -rw-r--r-- |
| checkpoint-47.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 25 分 2... | -rw-r--r-- |
| checkpoint-48.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 25 分 3... | -rw-r--r-- |
| checkpoint-49.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 25 分 4... | -rw-r--r-- |
| last.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 25 分 4... | -rw-r--r-- |
| model_best.pth.tar | 1,063,524,714 | tar-檔案 | 2025/11/02 11時 23 分 51... | -rw-r--r-- |
| summary.csv | 4,992 | csv-檔案 | 2025/11/02 11時 25 分 4... | -rw-r--r-- |

| epoch | train_loss | eval_loss | eval_top1 | eval_top5 | lr | |
|-------|-------------|-------------|-------------|-------------|-------------------|--|
| 29 | 93275759297 | 97623067502 | 1830985915 | 00000075219 | 0.00019 | |
| 30 | 56525957968 | 00155286385 | 73943661971 | 00000075219 | 0.00017 | |
| 31 | 61474738582 | 99669741240 | 9154929577 | 00000075219 | 0.00016 | |
| 32 | 10147726151 | 34971530000 | 9154929577 | 00000075219 | 0.00014 | |
| 33 | 06468450638 | 32231204616 | 57605633802 | 00000075219 | 0.00013 | |
| 34 | 10380430375 | 90389075749 | 33098591549 | 00000075219 | 0.00012 | |
| 35 | 94430145140 | 18707118235 | 43661971830 | 00000075219 | 0.00010 | |
| 36 | 90043107924 | 13520266304 | 71830985915 | 00000075219 | 400256282756e-05 | |
| 37 | 95216924144 | 33140711045 | 91549295774 | 00000075219 | 322351782782e-05 | |
| 38 | 62790941422 | 68353428635 | 57605633802 | 00000075219 | 784314464717e-05 | |
| 39 | 91879584713 | 80426010612 | 28169014084 | 00000075219 | 168930605272e-05 | |
| 40 | 30899800023 | 50969154734 | 67605633802 | 00000075219 | 751406263163e-05 | |
| 41 | 04248989782 | 00474713553 | 71830985915 | 00000075219 | 801862449629e-05 | |
| 42 | 31377379355 | 73024849152 | 9154929577 | 00000075219 | 329989034106e-05 | |
| 43 | 96695215471 | 39459486410 | 43661971830 | 00000075219 | 523688349516e-05 | |
| 44 | 74209628566 | 16685565088 | 71830985915 | 00000075219 | 5878527937163e-05 | |
| 45 | 45923326861 | 35051854895 | 47887323943 | 00000075219 | 870926211617e-05 | |
| 46 | 08767367947 | 21893964680 | 62.50000 | 00000075219 | 209717842259e-06 | |
| 47 | 41567130242 | 20514259875 | 57605633802 | 00000075219 | 873178278196e-06 | |
| 48 | 15782737730 | 01688008912 | 57605633802 | 00000075219 | 246713805587e-06 | |
| 49 | 85744362492 | 07067082633 | 57605633802 | 00000075219 | 78929321103e-07 | |

4. Slurm file explanations

- We merge the following two key parts in one Slurm file
 - Slurm script
 - Singularity command
- Some people put the Singularity command as an external shell script called by sbatch script.

4.1 Slurm script explanations

- Slurm (the workload manager) provides **sbatch**, **srun**, and **salloc** for submitting and running jobs on a cluster.
- **sbatch**: It submits a batch job script to Slurm's scheduler.
- **srun**: srun launches tasks (parallel steps) within a job or allocation. The terminal should be always online; or the training will be stopped.

4.1 Slurm directives

- We explain key parts in this section.
- --partition: dev, normal, and normal2
- dev has less resources, but could be scheduled quickly.
- Please read the table below.
- Change to normal later.

```
#!/bin/bash
#SBATCH --job-name=cls-sing
#SBATCH --partition=dev
#SBATCH --account=MSTXXX    ## iService_ID
## 選一種 GPU 申請方式（依叢集設定）
# 新版寫法（若支援）
##SBATCH --gpus=1
# 通用寫法（多數中心可用）
#SBATCH --gres=gpu:1
#SBATCH --cpus-per-task=4
#SBATCH --time=01:00:00
```

| 佇列 | 每個計劃最多 可用 GPU 總數 | 每個 Job 最大 執行時間 | 同時執行 job 數上限 | 佇列等候（排 隊）數上限 | GPU 型號 |
|---------|---------------------|-------------------|-----------------|-----------------|--------|
| dev | 8 | 2 小時 | 2 | 2 | H100 |
| normal | 16 | 48 小時 | 2 | 2 | H100 |
| normal2 | 16 | 48 小時 | 2 | 2 | H200 |

4.1 Slurm directives

- --gpus: Number of GPU 1 to 8
- -- gres: Number of GPU 1 to 8
- --cpus-per-task: CPU cores
- --time could be changed later.

```
#!/bin/bash
#SBATCH --job-name=cls-sing
#SBATCH --partition=dev
#SBATCH --account=MSTXXXX ## iService_ID
## 選一種 GPU 申請方式（依叢集設定）
# 新版寫法（若支援）
##SBATCH --gpus=1
# 通用寫法（多數中心可用）
#SBATCH --gres=gpu:1
#SBATCH --cpus-per-task=4
#SBATCH --time=01:00:00
#SBATCH --output=%x-%j.out
#SBATCH --error=%x-%j.err
```


4.2 User-tunable variables

- **CONTAINER**="/work/hpc_sys/sifs/pytorch_23.11-py3.sif" – Path to the Singularity container image used for the job. The details are shown in **Section 4.2.1** to **4.2.3**.
- **DATA_DIR**="\${HOME}/NCHC-Slurm-Demo/data/diabetic-retinopathy" – Directory containing training/validation/test data.
- **SRC_DIR**="\${HOME}/" – Base source directory where your code/repos live.
- **TIMM_DIR**="\${SRC_DIR}/NCHC-Slurm-Demo/pytorch-image-models" – Path to the local timm (pytorch-image-models) source tree.
- **EPOCHS**=50 – Number of training epochs.
- **IMG_H**=512, **IMG_W**=512 – Input image height and width fed to the model.
- **BATCH**=64 – Training batch size.
- **WORKERS**=\${SLURM_CPUS_PER_TASK:-4} – Number of DataLoader worker processes (defaults to cpus-per-task).

4.2.1 Singularity Container Image

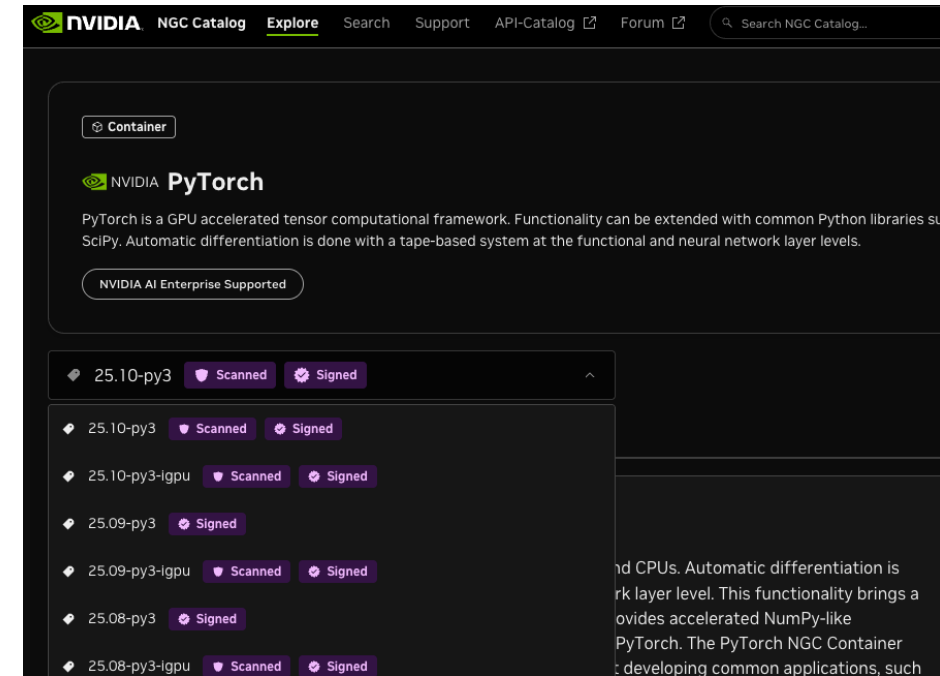
- CONTAINER environment is available from `"/work/hpc_sys/sifs/pytorch_23.11-py3.sif"`
- How to select other SIF files?
- **Method 1:** Obtain other older SIFs under `"/work/hpc_sys/sifs/"`.
 - `pytorch_22.05-py3.sif`
 - `pytorch_22.09-py3.sif`
 - `pytorch_22.09-py3_horovod.sif`
 - `pytorch_22.11-py3.sif`
 - `pytorch_23.02-py3_horovod.sif`

4.2.1 Singularity Container Image

- **Method 2: Apply the nVidia NGC**
- Step 1: Find the exact tag you want
- PyTorch and Tensorflow images packed by nVidia

<https://catalog.ngc.nvidia.com/orgs/nvidia/containers/pytorch>

<https://catalog.ngc.nvidia.com/orgs/nvidia/containers/tensorflow>



4.2.1 Singularity Container Image

- Step 2: Copy the image path

The screenshot shows the NVIDIA PyTorch container image page. The main header includes the NVIDIA logo and the text "PyTorch". Below this, a description states: "PyTorch is a GPU accelerated tensor computational framework. Functionality can be extended with common Python libraries such as NumPy and SciPy. Automatic differentiation is done with a tape-based system at the functional and neural network layer levels." A button labeled "NVIDIA AI Enterprise Supported" is visible. On the right, a "Get Container" button is present. The main content area displays the tag "25.10-py3" with "Scanned" and "Signed" badges. Below the tag, a navigation bar includes "Overview", "Tags", "Layers", "Security Scanning", and "Related Collections". A modal window is open, showing the tag "25.10-py3" and the instruction "Copy the image path for this tag below:". The image path "nvcrl.io/nvidia/pytorch:25.10-py3" is displayed in a text box, highlighted with a red border. Below the text box, there is a "View all tags" link and a "Publisher" label.

Container

NVIDIA **PyTorch**

PyTorch is a GPU accelerated tensor computational framework. Functionality can be extended with common Python libraries such as NumPy and SciPy. Automatic differentiation is done with a tape-based system at the functional and neural network layer levels.

NVIDIA AI Enterprise Supported

Get Container ▾

25.10-py3 Signed Scanned

Copy the image path for this tag below:

`nvcrl.io/nvidia/pytorch:25.10-py3`

[View all tags](#)

Publisher

4.2.1 Singularity Container Image

- Step 3: Pull the image by singularity (Called apptainer since 2021) on Nano5
- # General pattern
- `singularity pull pytorch_<TAG>.sif docker://nvcr.io/nvidia/pytorch:<TAG>`
- # Examples
- `singularity pull pytorch_25.10-py3.sif docker://nvcr.io/nvidia/pytorch:25.10-py3`

4.2.2 Build your own Container Image

- NGC container sometimes doesn't install the package we need, such as the libGL is used in YOLO.
- If you build the image including the required package, it should save time and budget.
- How to build the image is out of the scope of this slides.

4.3 Env / runtime settings

- module load singularity – Loads Singularity/Apptainer module so container commands work.
- export OMP_NUM_THREADS=\${WORKERS} – Sets OpenMP thread count (used by many math libs).
- export MKL_NUM_THREADS=\${WORKERS} – Sets number of threads for Intel MKL operations.
- export PATH="\${HOME}/.local/bin:\${PATH}" – Ensures user-installed Python tools in ~/.local/bin are found.

4.4 srun + container execution

- `srun singularity exec --nv` – Run the job under Slurm inside the container with GPU access enabled.
- `-B "${HOME}:${HOME}"` – Bind-mounts your home directory into the container.
- `--env`
`DATA_DIR=...,EPOCHS=...,IMG_H=...,IMG_W=...,BATCH=...,WORKERS=...` – Passes training hyperparameters into the container environment.

4.5 Adapt to another task

- You might like to train the object detection or LLM
- User variables section: Change the source code, dataset path, hyper parameters (epoch, batch size, and so on).
- Code setup inside the container: Replace the “install timm + basic libs” part with the dependent libraries.
- Training command: Swap the train.py (timm) invocation to your training script, passing specific options (e.g., model config, data yaml, image size).

5. Conclusions

- A good starting point to run algorithms on NCHC, particularly the deep learning on the Nano5.
- Nano5 could provide a great value for the deep learning research
- We could reduce the fee to buy the GPU which is not always run 365 days.
- This platform enhances the speed, flexibility, and our imaginations.

5.1 What is the next?

- Run your algorithms and dataset
- Enable the automatic mixed precision (AMP) during the forward pass during training.
- Use the Mobile App OTP Push instead of the Email OTP