

Lab5 FlashAttention

Nov, 2024 Parallel Programming

Overview

- ❖ Platform Guide (NCHC CT)
- ❖ Attention
- ❖ FlashAttention
- ❖ Lab5 Assignment

Platform Guide (NCHC CT)

NCHC Container

- ❖ Webpage: <https://portal.apps.edu-cloud.nchc.org.tw>
- ❖ Register your account first
- ❖ GPU: RTX 3070
- ❖ Available time: Tuesday, Wednesday and Sunday 00:00-23:59
- ❖ Total available GPUs: 46
- ❖ Please stop your container if you aren't using it

Register

1



NCHC.ai 國家高速網路與計算中心
National Center for High-performance Computing

課程介紹 基礎課程

登入 / 註冊



歡迎來到NCHC.ai

 Sign in with Google

OR

學生註冊

管理員 / 教師申請

By continuing, you agree to NCHC.ai's Terms of Service, Privacy Policy

2



歡迎註冊 NCHC.ai 帳戶

Your gmail account

* 帳號

請輸入您欲註冊的信箱

* 中文姓名

請輸入中文姓名

密碼

請輸入您的密碼

密碼確認

請再次輸入您的密碼

註冊 取消


Enter whatever
you want

Login

NCHC.ai 國家高速網路與計算中心
National Center for High-performance Computing

[課程介紹](#)[基礎課程](#)

登入 / 註冊



歡迎來到NCHC.ai

 Sign in with Google

OR

學生註冊

管理員 / 教師申請

By continuing, you agree to NCHC.ai's Terms of Service, Privacy Policy

Start Container

 陳凱揚
開課老師

開課列表

教室列表 1

工作清單

個人資料

密碼設定

登出

教室列表

113-1-清大-資工系-周志遠老師-平行程式

教課講師： kychen@lsalab.cs.nthu.edu.tw

學生人數： 119 位

| 課程名稱 | 課程程度 | 建立時間 | 操作 |
|----------------|------|----------------|---|
| NTHU-PP24-Hw4 | 進階 | 2024 / 11 / 14 |  開始 + 2 |
| NTHU-PP24-Lab5 | 基礎 | 2024 / 11 / 13 | |

 陳凱揚
開課老師

開課列表

教室列表

工作清單 3

個人資料

密碼設定

登出

工作清單

容器課程

NTHU-PP24-Lab5

 X

● 已開啟

0c09aa5b-0855-40ab-9b02-82a98ccb3797

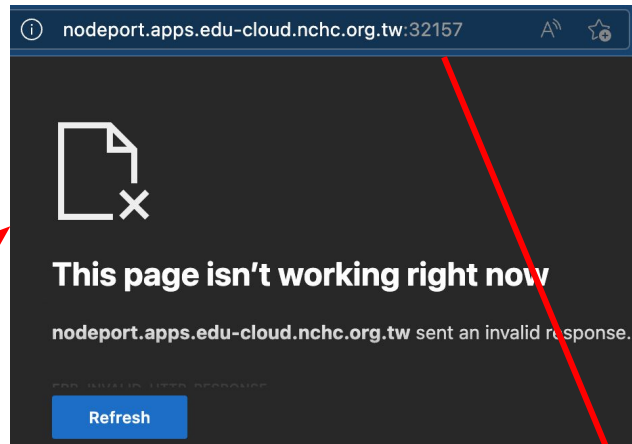
ssh | vscode

Access the Container

- SSH
 - The port number will be different
- Codeserver (the web version of vscode)
 - Click the “vscode” button



NTHU-PP24-Lab5

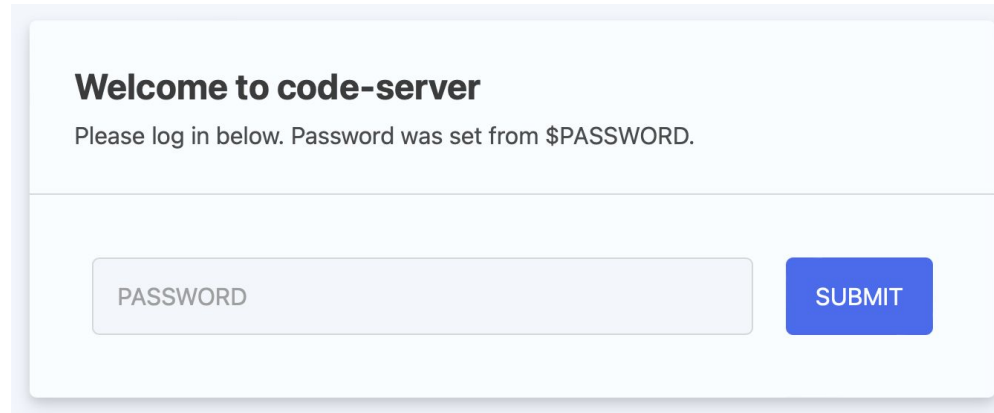


MobaXterm or Terminal:

ssh [root@nodeport.apps.edu-cloud.nchc.org.tw](https://nodeport.apps.edu-cloud.nchc.org.tw) -p 32157

User & Password

- User: root
- Password: student
 - The password of `ssh` and `code-server` is the same



The image shows a login interface for code-server. It has a light blue header with the text 'Welcome to code-server' and a message 'Please log in below. Password was set from \$PASSWORD.' Below this is a white input field with the placeholder text 'PASSWORD' and a blue 'SUBMIT' button.

Welcome to code-server

Please log in below. Password was set from \$PASSWORD.

PASSWORD

SUBMIT

First-time Setup Script

Open terminal in the container:

```
bash <(curl -s https://apollo.cs.nthu.edu.tw/pp24/share/script/setup-remote.sh)
```

The script will execute the following commands:

- Set proper **bash** config for homework judge (e.g., hw4-remote-judge)
- Generate ssh key and install it on Apollo (you will be prompted to enter your Apollo account name and password)

Run this script only **once**, even you relaunched your container (since your personal data will be kept).

```
#####
##      Installing SSH Key.      ##
#####
Enter your username on apollo: pp24s085
/root/.ssh/id_rsa already exists.
Overwrite (y/n)? /usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
The authenticity of host 'apollo-gpu.cs.nthu.edu.tw (140.114.91.189)' can't be established.
ED25519 key fingerprint is SHA256:5cV2dr2KzW4Iupn5g1e4xe+b0KV1T0UF1dEim0ISCp4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
(pp24s085@apollo-gpu.cs.nthu.edu.tw) Password:
Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'pp24s085@apollo-gpu.cs.nthu.edu.tw'"
and check to make sure that only the key(s) you wanted were added.

% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           %         %         Dload  Upload    Total   Spent    Left   Speed
100 3992    100 3992     0     0  48487      0 --:--:-- --:--:-- --:--:-- 48682
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           %         %         Dload  Upload    Total   Spent    Left   Speed
100 18584   100 18584     0     0   216k      0 --:--:-- --:--:-- --:--:-- 218k
#####
##      Install completed      ##
## Please relogin the container ##
#####
root@PP24:~#
```

Stop your container

- Please stop your container if you aren't using it; otherwise, other students may not have enough GPU resources
- Your files located under \$HOME (/root/) will be preserved



NTHU-PP24-Lab5



NO MINING

Educational use only

Please cherish the computing resources we provided

Attention

Attention

$$\text{❖ } \text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad \mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{N \times d}$$

- ❖ Q: What we're focusing on.
- ❖ K: What features are available.
- ❖ V: What content is retrieved based on focus.

Attention

$$\mathbf{S} = \mathbf{Q}\mathbf{K}^\top \in \mathbb{R}^{N \times N}, \quad \mathbf{P} = \text{softmax}(\mathbf{S}) \in \mathbb{R}^{N \times N}, \quad \mathbf{O} = \mathbf{P}\mathbf{V} \in \mathbb{R}^{N \times d},$$

$$\begin{bmatrix} s_{11} & s_{12} & s_{13} & s_{14} \\ s_{21} & s_{22} & s_{23} & s_{24} \\ s_{31} & s_{32} & s_{33} & s_{34} \\ s_{41} & s_{42} & s_{43} & s_{44} \end{bmatrix} = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{32} & q_{33} \\ q_{41} & q_{42} & q_{43} \end{bmatrix} \cdot \begin{bmatrix} k_{11} & k_{21} & k_{31} & k_{41} \\ k_{12} & k_{22} & k_{32} & k_{42} \\ k_{13} & k_{23} & k_{33} & k_{43} \end{bmatrix}$$

Attention

$$\mathbf{S} = \mathbf{Q}\mathbf{K}^\top \in \mathbb{R}^{N \times N}, \quad \mathbf{P} = \text{softmax}(\mathbf{S}) \in \mathbb{R}^{N \times N}, \quad \mathbf{O} = \mathbf{P}\mathbf{V} \in \mathbb{R}^{N \times d},$$

$$\begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \\ p_{41} & p_{42} & p_{43} & p_{44} \end{bmatrix} = \text{softmax} \left(\begin{bmatrix} s_{11} & s_{12} & s_{13} & s_{14} \\ s_{21} & s_{22} & s_{23} & s_{24} \\ s_{31} & s_{32} & s_{33} & s_{34} \\ s_{41} & s_{42} & s_{43} & s_{44} \end{bmatrix} \right)$$

Attention

$$\mathbf{S} = \mathbf{Q}\mathbf{K}^\top \in \mathbb{R}^{N \times N}, \quad \mathbf{P} = \text{softmax}(\mathbf{S}) \in \mathbb{R}^{N \times N}, \quad \mathbf{O} = \mathbf{P}\mathbf{V} \in \mathbb{R}^{N \times d};$$

$$\begin{bmatrix} o_{11} & o_{12} & o_{13} \\ o_{21} & o_{22} & o_{23} \\ o_{31} & o_{32} & o_{33} \\ o_{41} & o_{42} & o_{43} \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \\ p_{41} & p_{42} & p_{43} & p_{44} \end{bmatrix} \cdot \begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \\ v_{31} & v_{32} & v_{33} \\ v_{41} & v_{42} & v_{43} \end{bmatrix}$$

Multi-Head Attention

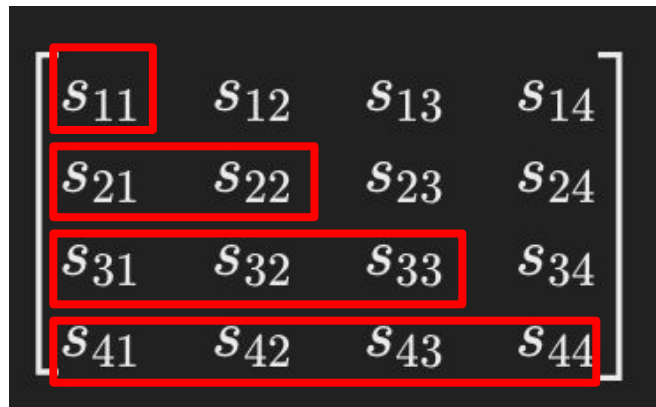
- ❖ Rich Representations
- ❖ Efficient Parallelization
- ❖ E.g. `emb_dim = 4096` -> `num_heads = 32`, `head_size = 128`

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O$$

where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$

Causal Attention

- ❖ If you're predicting the next word in a sentence, the model shouldn't have access to future words beyond the current position.



| | | | |
|----------|----------|----------|----------|
| s_{11} | s_{12} | s_{13} | s_{14} |
| s_{21} | s_{22} | s_{23} | s_{24} |
| s_{31} | s_{32} | s_{33} | s_{34} |
| s_{41} | s_{42} | s_{43} | s_{44} |

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V$$

$$\text{Masked Scores}_{i,j} = \begin{cases} \frac{(QK^T)_{i,j}}{\sqrt{d_k}}, & \text{if } j \leq i \\ -\infty, & \text{if } j > i \end{cases}$$

$$\text{Masked Attention Weights}_{i,j} = \text{softmax}(\text{Masked Scores}_{i,j})$$

$$\text{Causal Attention}(Q, K, V) = \text{softmax} \left(\text{Mask} \left(\frac{QK^T}{\sqrt{d_k}} \right) \right) V$$

FlashAttention

FlashAttention - Overview

- ❖ Goal: avoid reading and writing the attention matrix to and from HBM.
 - Computing the softmax without access to the whole input.
 - Not storing the large intermediate attention matrix for the backward pass.

Algorithm 0 Standard Attention Implementation

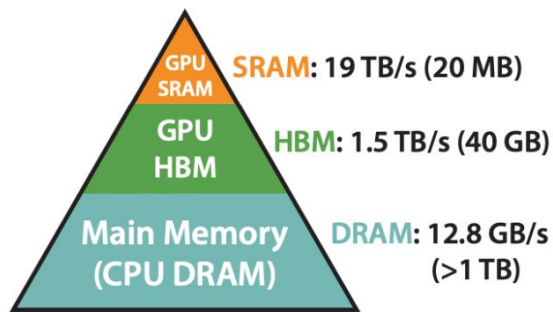
Require: Matrices $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{N \times d}$ in HBM.

- 1: Load \mathbf{Q}, \mathbf{K} by blocks from HBM, compute $\mathbf{S} = \mathbf{QK}^\top$, write \mathbf{S} to HBM.
 - 2: Read \mathbf{S} from HBM, compute $\mathbf{P} = \text{softmax}(\mathbf{S})$, write \mathbf{P} to HBM.
 - 3: Load \mathbf{P} and \mathbf{V} by blocks from HBM, compute $\mathbf{O} = \mathbf{PV}$, write \mathbf{O} to HBM.
 - 4: Return \mathbf{O} .
-

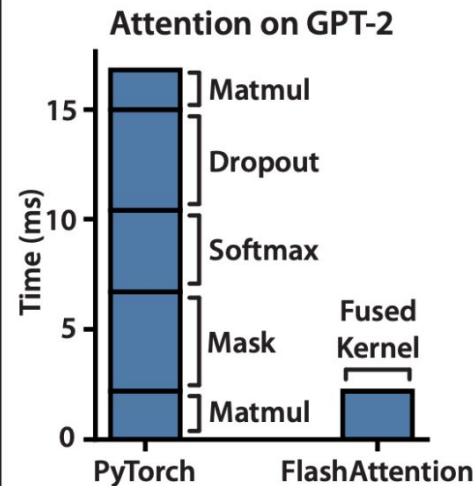
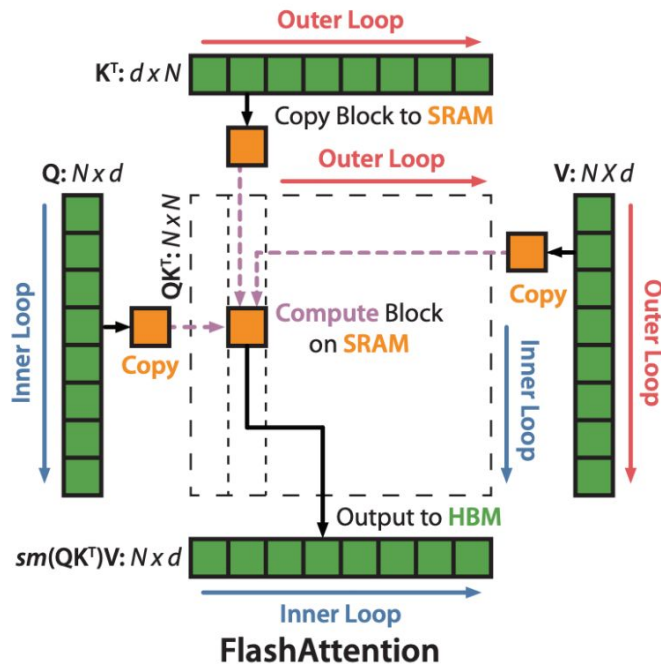
FlashAttention - Method

- ❖ Tiling: split the input into blocks and make several passes over input blocks.
 - Matrix multiplication and pointwise operations are easy to handle.
 - SoftMax: need to maintain $m(x)$, $l(x)$.
- ❖ Recompute: store the softmax normalization factor in order to quickly recompute in the backward pass.

FlashAttention



Memory Hierarchy with Bandwidth & Memory Size



FlashAttention - SoftMax

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^d e^{x_j}}$$

$$m = \max_i (x_i); \quad \text{softmax}(x_i) = \frac{e^{x_i - m}}{\sum_{j=1}^d e^{x_j - m}}$$

FlashAttention - SoftMax

$$m(x) := \max_i x_i, \quad f(x) := [e^{x_1 - m(x)} \quad \dots \quad e^{x_B - m(x)}], \quad \ell(x) := \sum_i f(x)_i, \quad \text{softmax}(x) := \frac{f(x)}{\ell(x)}.$$

For vectors $x^{(1)}, x^{(2)} \in \mathbb{R}^B$, we can decompose the softmax of the concatenated $x = [x^{(1)} \ x^{(2)}] \in \mathbb{R}^{2B}$ as:

$$m(x) = m([x^{(1)} \ x^{(2)}]) = \max(m(x^{(1)}), m(x^{(2)})), \quad f(x) = [e^{m(x^{(1)}) - m(x)} f(x^{(1)}) \quad e^{m(x^{(2)}) - m(x)} f(x^{(2)})],$$
$$\ell(x) = \ell([x^{(1)} \ x^{(2)}]) = e^{m(x^{(1)}) - m(x)} \ell(x^{(1)}) + e^{m(x^{(2)}) - m(x)} \ell(x^{(2)}), \quad \text{softmax}(x) = \frac{f(x)}{\ell(x)}.$$

FlashAttention - SoftMax

$$m_1 = \max([1, 2]) = 2$$

$$m_2 = \max([3, 4]) = 4$$

$$m = \max(m_1, m_2) = 4$$

$$f_1 = [e^{1-2}, e^{2-2}] = [e^{-1}, e^0]$$

$$f_2 = [e^{3-4}, e^{4-4}] = [e^{-1}, e^0]$$

$$f = [e^{m_1-m} f_1, e^{m_2-m} f_2] = [e^{-3}, e^{-2}, e^{-1}, e^0]$$

$$l_1 = \sum f_1 = e^{-1} + e^0$$

$$l_2 = \sum f_2 = e^{-1} + e^0$$

$$l = e^{m_1-m} l_1 + e^{m_2-m} l_2 = e^{-3} + e^{-2} + e^{-1} + e^0$$

$$o_1 = \frac{f_1}{l_1} = \frac{[e^{-1}, e^0]}{e^{-1} + e^0}$$

$$o_2 = \frac{f_2}{l_2} = \frac{[e^{-1}, e^0]}{e^{-1} + e^0}$$

$$o = \frac{f}{l} = \frac{[e^{-3}, e^{-2}, e^{-1}, e^0]}{e^{-3} + e^{-2} + e^{-1} + e^0}$$

FlashAttention - Algorithm

Algorithm 1 FLASHATTENTION

Require: Matrices $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{N \times d}$ in HBM, on-chip SRAM of size M .

- 1: Set block sizes $B_c = \lceil \frac{M}{4d} \rceil, B_r = \min(\lceil \frac{M}{4d} \rceil, d)$.
 - 2: Initialize $\mathbf{O} = (0)_{N \times d} \in \mathbb{R}^{N \times d}, \ell = (0)_N \in \mathbb{R}^N, m = (-\infty)_N \in \mathbb{R}^N$ in HBM.
 - 3: Divide \mathbf{Q} into $T_r = \lceil \frac{N}{B_r} \rceil$ blocks $\mathbf{Q}_1, \dots, \mathbf{Q}_{T_r}$ of size $B_r \times d$ each, and divide \mathbf{K}, \mathbf{V} into $T_c = \lceil \frac{N}{B_c} \rceil$ blocks $\mathbf{K}_1, \dots, \mathbf{K}_{T_c}$ and $\mathbf{V}_1, \dots, \mathbf{V}_{T_c}$, of size $B_c \times d$ each.
 - 4: Divide \mathbf{O} into T_r blocks $\mathbf{O}_1, \dots, \mathbf{O}_{T_r}$ of size $B_r \times d$ each, divide ℓ into T_r blocks $\ell_1, \dots, \ell_{T_r}$ of size B_r each, divide m into T_r blocks m_1, \dots, m_{T_r} of size B_r each.
 - 5: **for** $1 \leq j \leq T_c$ **do**
 - 6: Load $\mathbf{K}_j, \mathbf{V}_j$ from HBM to on-chip SRAM.
 - 7: **for** $1 \leq i \leq T_r$ **do**
 - 8: Load $\mathbf{Q}_i, \mathbf{O}_i, \ell_i, m_i$ from HBM to on-chip SRAM.
 - 9: On chip, compute $\mathbf{S}_{ij} = \mathbf{Q}_i \mathbf{K}_j^T \in \mathbb{R}^{B_r \times B_c}$.
 - 10: On chip, compute $\tilde{m}_{ij} = \text{rowmax}(\mathbf{S}_{ij}) \in \mathbb{R}^{B_r}, \tilde{\mathbf{P}}_{ij} = \exp(\mathbf{S}_{ij} - \tilde{m}_{ij}) \in \mathbb{R}^{B_r \times B_c}$ (pointwise), $\tilde{\ell}_{ij} = \text{rowsum}(\tilde{\mathbf{P}}_{ij}) \in \mathbb{R}^{B_r}$.
 - 11: On chip, compute $m_i^{\text{new}} = \max(m_i, \tilde{m}_{ij}) \in \mathbb{R}^{B_r}, \ell_i^{\text{new}} = e^{m_i - m_i^{\text{new}}} \ell_i + e^{\tilde{m}_{ij} - m_i^{\text{new}}} \tilde{\ell}_{ij} \in \mathbb{R}^{B_r}$.
 - 12: Write $\mathbf{O}_i \leftarrow \text{diag}(\ell_i^{\text{new}})^{-1} (\text{diag}(\ell_i) e^{m_i - m_i^{\text{new}}} \mathbf{O}_i + e^{\tilde{m}_{ij} - m_i^{\text{new}}} \tilde{\mathbf{P}}_{ij} \mathbf{V}_j)$ to HBM.
 - 13: Write $\ell_i \leftarrow \ell_i^{\text{new}}, m_i \leftarrow m_i^{\text{new}}$ to HBM.
 - 14: **end for**
 - 15: **end for**
 - 16: Return \mathbf{O} .
-

Reference

- ❖ <https://arxiv.org/pdf/2205.14135>
- ❖ <https://www.cvmart.net/community/detail/7943>
- ❖ <https://www.youtube.com/watch?v=eMlx5fFNoYc>

Lab5 Assignment

Objective

- ❖ Evaluate the performance of attention mechanisms by comparing:
 - The original PyTorch implementation
 - The FlashAttention v2 implementation
- ❖ Analyze the benefits of FlashAttention and explore its advantages over the standard approach.
- ❖ Conduct benchmarking with varying parameters and compare the results to gain deeper insights.

Benchmark Script

- ❖ TA provide the Python benchmark script that does not require any modifications.
- ❖ Your task is to adjust only the parameters within the benchmark.
- ❖ The result will be outputted to a JSON file.
 - Execution time
 - FLOPs
 - Peak memory usage

```
{} benchmark_result.json ×
lab5 > {} benchmark_result.json > ...
1  {
2    "forward": {
3      "time(s)": 0.007095515976349513,
4      "FLOPS(TFLOPs/s)": 38.739664297876566
5    },
6    "backward": {
7      "time(s)": 0.018877355754375456,
8      "FLOPS(TFLOPs/s)": 36.40312638599925
9    },
10   "forward_backward": {
11     "time(s)": 0.025972871730724968,
12     "FLOPS(TFLOPs/s)": 37.04144402199095
13   },
14   "peak_memory_usage(MB)": 1288.00048828125
15 }
```


Benchmark Script

- ❖ Test following parameters and compare the results.
 - **batch_size**: int
 - **seq_len**: int
 - **num_heads**: int, (must be divisible by emb_dim)
 - **emb_dim**: int
 - **impl**: str, (choose between Pytorch and Flash2)
 - **causal**: bool

Preparation

- ❖ File are Located at `/tmp/lab5` on **NCHC**.
- ❖ The benchmark script is named `lab5.py`.
- ❖ Use `python lab5.py --help` to view detailed parameter descriptions.
- ❖ Important Notes:
 - Do not modify the source code.
 - Do not merge multiple tests into a single run, as this may result in incorrect output values.
- ❖ To run multiple tests, you can use a bash script for automation.

Workflow

- ❖ `cp -r /tmp/lab5 ~/lab5 && cd ~/lab5`
- ❖ `python lab5.py \`
 - `--batch_size 32 \`
 - `--seq_len 1024 \`
 - `--num_heads 32 \`
 - `--emb_dim 2048 \`
 - `--impl Flash2 \`
 - `--causal \`
 - `--repeats 30 \`
 - `--output benchmark_result.json`

Submission

- ❖ Plot the experimental data in a chart for better visualization.
- ❖ Analyze and explain your observations based on the collected data.
- ❖ Submit your report as a `lab5.pdf` file to eeclass before **11/28 23:59**.
- ❖ Important Notes:
 - Get started as soon as possible since the NCHC platform is only accessible on Tuesday, Wednesday and Sunday.
 - Remember to stop your container when not in use.