PyTorch Seminar

Day3. Good Practices for Training Code

Duhyeon Kim / May 2025

PyTorch

PyTorch Seminar

Day3. Good Practices for Training Code

Duhyeon Kim / May 2025

PyTorch

Good Practices for Training Code

Overview

A. Practical Tools in Code: tqdm, device, argparse, compute_effi, tensorboard

B. Model Initialization Strategies

C. Schedulers and Checkpointing

D. Loss Tracking and Evaluation Stages

E. Additional Features

Practical Tools in Code

tqdm, device, argparse, compute_effi, tensorboard

tqdm

```
Processing: 100%|██████████| 100/100 [00:02<00:00, 41.06it/s]
```

device — CPU or GPU or XPU

argparse

```
(Command line) $ python train.py —mode train —epoch 100
```

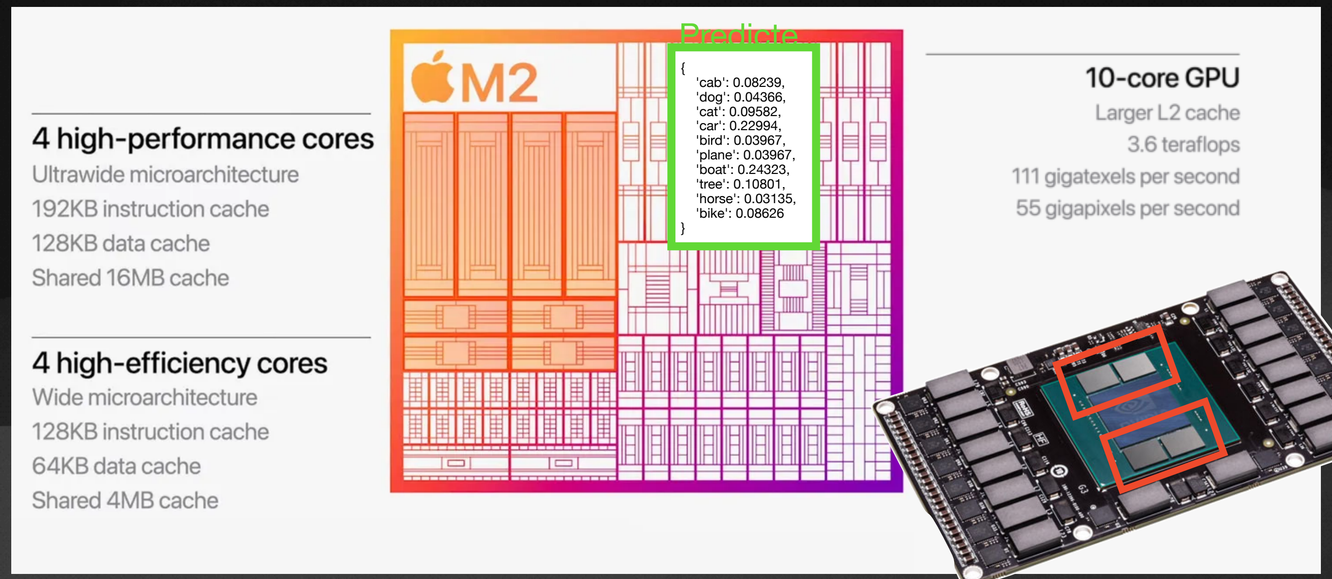
compute_efficiency — compute_time / (compute + read time)

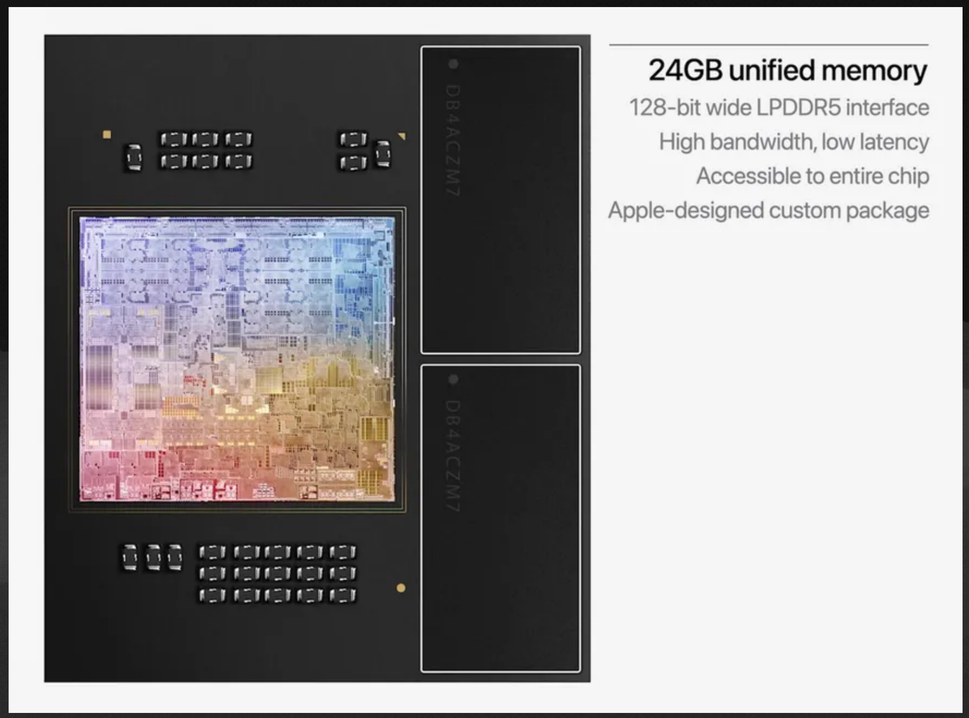
tensorboard — Visualize everything

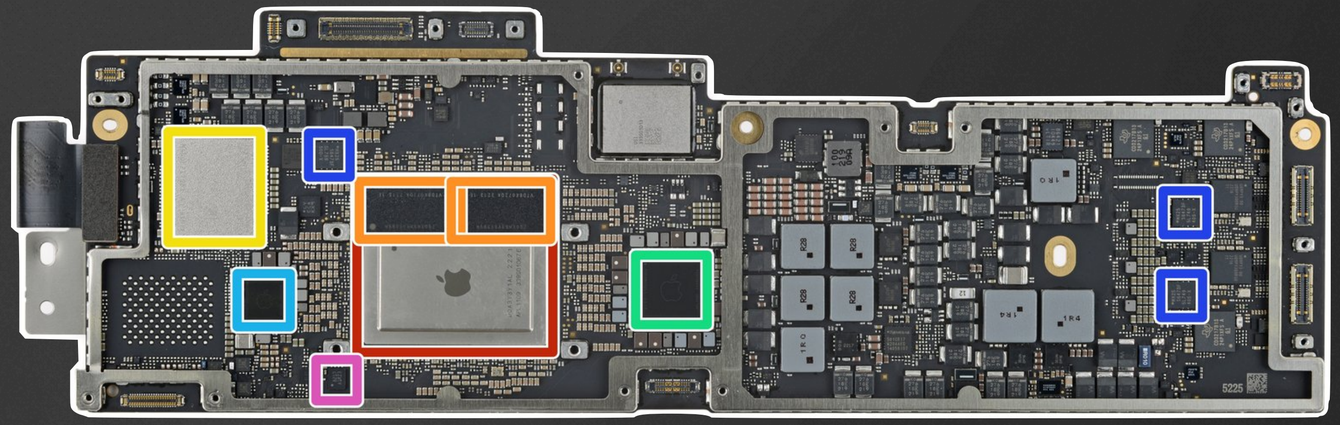
https://ko.ifixit.com/Guide/Macbook+Air+(M2+2022)+Logic+Board+and+Chip+Identification/151816
WWDC2022

Computational Efficiency ★

Computer Archi







1) ssd to ram

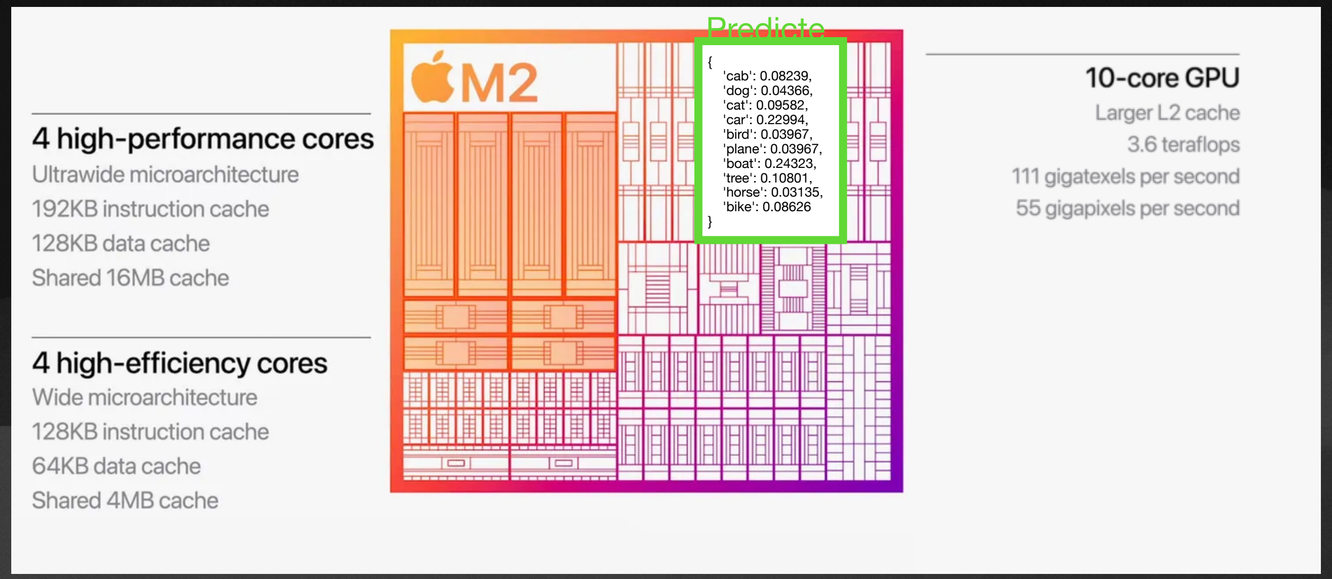
2) ram to GPU (gddr, hbm)

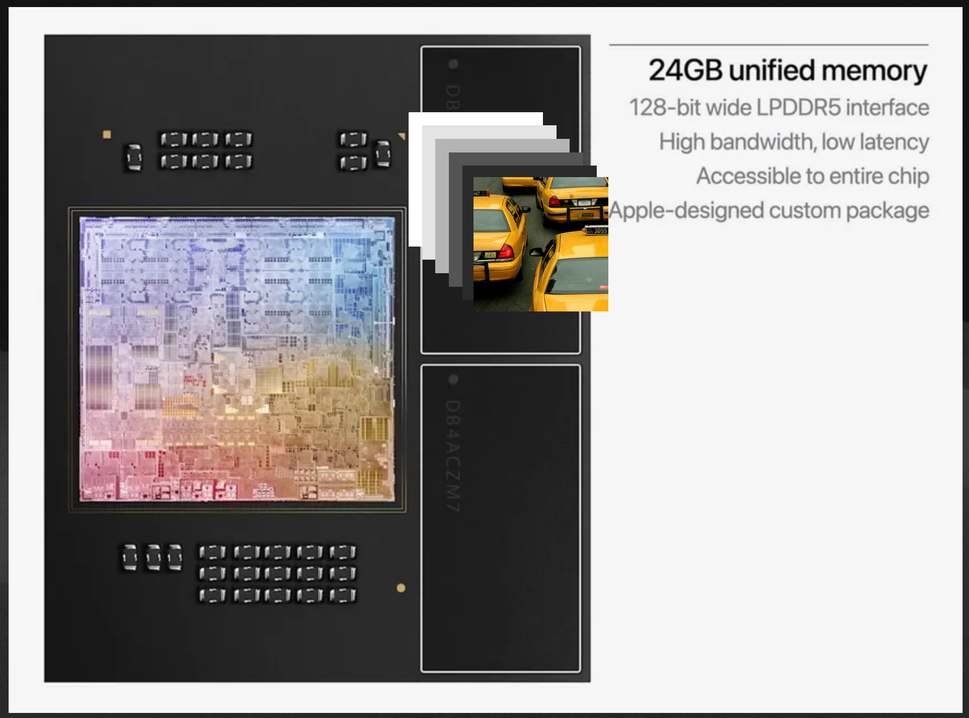
3) GPU computation

https://ko.ifixit.com/Guide/Macbook+Air+(M2+2022)+Logic+Board+and+Chip+Identification/151816
WWDC2022

Computational Efficiency ★

Computer Archi





1) ssd to ram

2) ram to GPU

3) GPU computation

$$\text{Compute Efficiency} = \frac{T_{\text{gpu\_compute}}^{(3)}}{T_{\text{ssd2ram}}^{(1)} + T_{\text{ram2gpu}}^{(2)} + T_{\text{gpu\_compute}}^{(3)}}$$

Schedulers

Learning rate Scheduling

Using a fixed learning rate throughout training often leads to suboptimal results: if the rate is too high, the model may oscillate or diverge; if too low, training becomes slow and may get stuck in poor local minima

Key Concepts:

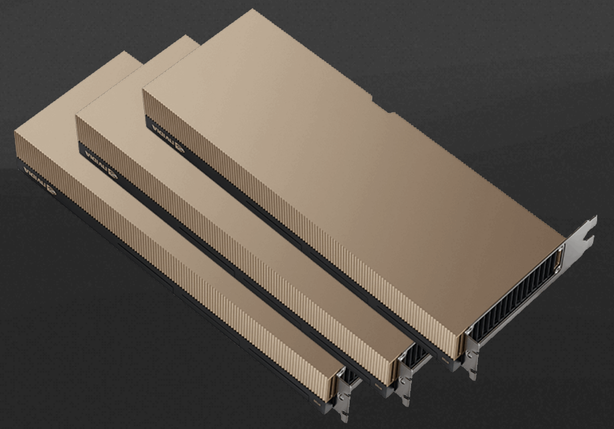
- Rapid Progress Early, Fine-Tuning Later
- Avoiding Poor Convergence
- Faster and More Stable Training
- Adaptive to Training Dynamics

Schedulers

Learning rate Scheduling

Key Concepts:

- Rapid Progress Early, Fine-Tuning Later
- Avoiding Poor Convergence
- Faster and More Stable Training
- Adaptive to Training Dynamics



If you don’t have an H100, do everything you can—time is money!

https://docs.pytorch.org/docs/stable/optim.html#how-to-adjust-learning-rate

Checkpoints

Saving Model Params

For training Model, we have defined and updated:

- Model Parameters + Model Structure
- Optimizer & Learning Rate
- Scheduler
- Epochs

These things are saved in form of dictionary.

Checkpoints

Saving Model Params

These things are saved in form of dictionary.

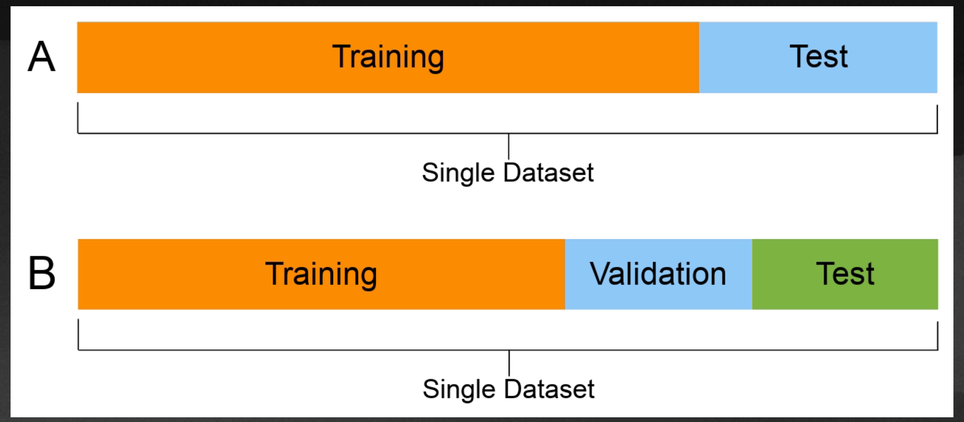
```
checkpoint = {
    'model_state_dict': model.state_dict(),
    'optimizer_state_dict': optimizer.state_dict(),
    'epoch': epoch,
    # 기타 정보
}
torch.save(checkpoint, 'checkpoint.pth')
```

```
model.load_state_dict(checkpoint['model_state_dict'])
optimizer.load_state_dict(checkpoint['optimizer_state_dict'])
```

While .pth files are used exclusively with PyTorch,
ONNX provides a universal graph format that can be used for
inference across various environments and frameworks.

Evaluation and Loss Tracking

Train, Val, Test Loss



Train Loss *(Model trained)*
Validation Loss *(Validation)*
Test Loss *(Never-Seen data)*

Loss is a fundamental indicator that guides the optimization process and measures how well the model is learning from the data.

https://davincilabs.ai/blog/?bmode=view&idx=10640969

Model Initialization Strategies

normal, Xavier, kaiming, orthogonal

normal Init	mean, std	Simple MLP, Shallow Network
xavier Init	fan_in, fan_out, std	Network using Sigmoid, Tanh
kaiming Init	fan_in, negative slope, std	Net using ReLU, LeakyReLU (DL)
orthogonal Init	Orthogonal matrix, norm	RNN, Deep MLP

Good Practices for Training in Code

Assignment #2

Improve Your Training Code

Based on the code submitted in Assignment #1,
Apply at least 3 techniques from the Day 3 materials.

🧾 Submission Requirements

- 📁 Jupyter Notebook (.ipynb) - Enhanced code implementation
- 📋 Technical Report (PDF) - Detailed explanation of applied techniques

Thank you

Summary

Overall Code Structure

CNN

MLP

1 2 3 4 5 6 7 8 9 0

Predicted

```
{
    'cab': 0.08239,
    'dog': 0.04366,
    'cat': 0.09582,
    'car': 0.22994,
    'bird': 0.03967,
    'plane': 0.03967,
    'boat': 0.24323,
    'tree': 0.10801,
    'horse': 0.03135,
    'bike': 0.08626
}
```

Answer

```
{
    'cab': 1,
    'dog': 0,
    'cat': 0,
    'car': 0,
    'bird': 0,
    'plane': 0,
    'boat': 0,
    'tree': 0,
    'horse': 0,
    'bike': 0
}
```

Dataloader

CustomDataset

batches

Cross Entropy Loss

$$L = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

$$y_j = 1, \quad y_{i \neq j} = 0$$

$$L = -\log(\hat{y}_j)$$

SGD optimizer

$$\theta_{new} = \theta_{old} - \eta \cdot \nabla_\theta \cdot lr$$