

第三次实验说明

- [第三次实验说明](#)
 - [一、实验内容](#)
 - [二、实验环境配置](#)
 - [三、提交方式](#)
 - [四、补充材料（并非本次实验内容，感兴趣可做课后补充）](#)
 - [1. PyTorch教程](#)
 - [2. 使用GPU加快模型训练速度](#)

Written by: 助教-谢羿衡

对实验内容有疑问可通过QQ私信联系助教~

一、实验内容

本次实验是基于CIFAR-10数据集与ResNet残差神经网络的图像分类实验，由于实验内容较多，后两次实验课都做这一次实验

具体实验内容如下：

1. 基于CIFAR-10图像分类数据集，使用同样的训练设置，分别训练以下三个模型，并对训练得到的三个模型的性能进行对比分析：
 - 自主设计的ResNet74图像分类模型
 - PyTorch官方提供的ResNet50模型
 - PyTorch官方提供的ResNet101模型
2. 基于自主设计的ResNet74模型，将不同训练epoch得到的模型训练与测试精度进行可视化
3. 基于自主设计的ResNet74模型，设计实现至少包括亮度调整、随机噪声、色调调整、随机裁剪、随机翻转、Mixup的数据增强策略；并探讨分析不同数据增强策略对模型性能的影响
4. 基于自主设计的ResNet74模型，探索分别模型训练时，是否使用Dropout对模型性能的影响
5. 基于自主设计的ResNet74模型，量化对比分析采用SGD、Momentum、Adagrad、Adam等不同优化器对模型训练模型的影响
6. 基于自主设计的ResNet74模型，尝试采用不同学习率和学习策略进行模型训练，并对实验结果进行分析

二、实验环境配置

本次实验需要使用 `PyTorch`，这是一个功能强大且广泛应用的深度学习框架，能够便捷且高效地实现深度学习模型的训练与测试。

本次实验建议新建一个实验环境，具体教程请参考第一次实验说明。**「注意：最新版 PyTorch 需要 Python 3.10及更高版本」**

新建实验环境后，需要在该环境下安装 `torch`（PyTorch基础库）与 `torchvision`（PyTorch图像处理库）两个库。如果你的电脑有独立显卡，可以尝试[使用GPU版本的库](#)来加快训练速度

环境配置完成后即可创建 `.ipynb` 代码并开始实验。

本次实验内容较多，助教提供了示例代码 `ResNet50_CIFAR10.ipynb`，可供同学们参考。

三、提交方式

- 提交**「两个」**文件：
 - 代码文件（`.ipynb`）
 - 实验报告（`.pdf` / `.doc` / `.docx`），实验报告模板在群文件
- 文件命名为：`姓名-学号-实验三`，下面是例子：
 - `谢羿衡-2023302061163-实验三.ipynb`
 - `谢羿衡-2023302061163-实验三.pdf`
- 交到助教邮箱：`51729575@qq.com`，邮件主题为：`姓名-学号-实验三`
- 截止日期：`2025年12月21日 23:59前`

四、补充材料（并非本次实验内容，感兴趣可做课后补充）

1. PyTorch教程

油管上的一个PyTorch教学视频，时长很长（25小时），但是内容非常全面。再也不用担心看不懂AI写的代码了 [PyTorch for Deep Learning & Machine Learning – Full Course](#)

「注：非常不建议在期末月观看」

2. 使用GPU加快模型训练速度

助教的电脑配有RTX 4060 Laptop GPU。在GPU上训练ResNet50网络, 一个epoch只需要30s; 而CPU(Intel i7-13700H)需要5min。

在开始之前，请先确认你的设备是否支持 GPU 加速，以及支持的CUDA版本。可以通过以下两个命令查看：

- `nvidia-smi` : 显示的是GPU驱动信息。12.7的意思是当前驱动程序支持的最高CUDA运行时版本是 12.7

```

PowerShell
> pwsh
>> nvidia-smi
Mon Dec 8 13:57:36 2025

+-----+
| NVIDIA-SMI 566.24                  Driver Version: 566.24          CUDA Version: 12.7          |
+-----+-----+-----+-----+-----+-----+
| GPU Name                               Driver-Model | Bus-Id          Disp.A | Volatile Uncorr. ECC | |
| Fan  Temp  Perf              Pwr:Usage/Cap |                | Memory-Usage | GPU-Util  Compute M. |
|                                           |                | MIG M.       |
+-----+-----+-----+-----+-----+-----+
|    0  NVIDIA GeForce RTX 4060 ... WDDM | 00000000:01:00.0 On |          | N/A | |
| N/A   46C    P8              5W / 125W | 2244MiB / 8188MiB |          | 6%   | Default |
|                                           |                | MIG M.       | N/A |
+-----+-----+-----+-----+-----+-----+

```

- `nvcc --version`：显示的是CUDA Toolkit的版本信息。CUDA Toolkit包含了开发工具和库，用于编译和运行 CUDA 程序（「PyTorch会调用Toolkit提供的库」）。13.1的意思工具包支持的最高CUDA运行时版本是13.1

```
> pwsh 0ms
>> nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2025 NVIDIA Corporation
Built on Fri_Nov__7_19:25:04_Pacific_Standard_Time_2025
Cuda compilation tools, release 13.1, V13.1.80
Build cuda_13.1 r13.1/compiler.36836380_0
```

(上面是便于理解的说法，实际上不太严谨，[这里](#)是一个相对完整且严谨的补充)

CUDA版本是「**向后兼容**」的，也就是说高版本能兼容低版本。上面两张图说明这个电脑能用的CUDA版本最高是12.7。

查看完CUDA版本后，就可以去[PyTorch官网](#)获取安装命令。你需要根据自己电脑上的情况选择不同的安装项。

PyTorch Build	Stable (2.9.1)		Preview (Nightly)		
Your OS	Linux	Mac	Windows		
Package	Pip	LibTorch	Source		
Language	Python		C++ / Java		
Compute Platform	CUDA 12.6	CUDA 12.8	CUDA 13.0	ROCm 6.4	CPU
Run this Command:	pip3 install torch torchvision --index-url https://download.pytorch.org/whl/cu126				

如需安装旧版本PyTorch，请查看[此处](#)

可以使用下面的代码测试GPU版本的库是否安装成功：

```
import torch
# Check for GPU
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
print(f"Using device: {device}")
```

如安装成功，会显示：

```
Using device: cuda
```