

2024 Fall - Machine Learning Assignment

目录

- [介绍](#)
- [项目结构](#)
- [作业内容](#)
- [提交要求](#)
- [评估与学术诚信](#)

介绍

在本次作业中，你将通过实际操作在CIFAR-10数据集上完成一个小项目。你将使用PyTorch框架完成模型的搭建，以及通过数据增强、正则化和学习率调度等技术提升模型的泛化能力。此外，你还将学习如何使用Wandb进行实验管理和超参数调优。

项目结构

以下是项目的文件结构及各文件的功能描述：

```
ML_Assignment/
├── README.md                # 项目说明文件
├── requirements.txt         # 依赖包列表
├── data/                   # 数据集相关文件夹
│   └── prepare_data.py     # 数据准备脚本
├── models/                 # 模型相关文件夹
│   ├── SimpleCNN.py        # 简易版卷积神经网络模型
│   ├── SimpleViT.py        # 简易版Vision Transformer模型
│   └── SimpleComposition.py # 简单的组合模型
├── model_ckpt/             # 模型检查点文件夹
├── scripts/                # 训练和测试脚本文件夹
│   ├── train.py            # 训练脚本
│   └── test.py             # 测试脚本
```

提交要求与评分依据

本次提交包括代码与报告两个部分。

- 代码部分：请你在最后运行 `python compress.py` 来完成代码的打包。注意，你的打包内容不包括自己训练得到的模型参数：我们将会在测试环境中对你的模型进行重新进行训练，以避免可能的数据泄露问题。此外，我们会对提交的代码进行查重，请勿复制他人的代码或进行轻微修改以规避查重系统。
- 报告部分：在以下作业内容中，`代码完成度` 与 `Wandb记录情况` 的相关评分均需体现在报告中。

1. 你需要将要求完成的相关code snippets以截图或代码文本环境的形式报告中，并且进行必要的阐述（例如，你所实现的相应的model architecture、你的训练基本流程、采用的optimizer、scheduler等）；合理且无抄袭雷同情况，即得满分。
2. 请你将Wandb中记录的每一个metric的曲线进行截图，放置在报告中，并进行简单分析；合理即得满分。

作业内容

在本次作业中，你只能在**TODO**标记的部分添加代码。请不要修改其他部分的代码，以避免潜在的测试问题。在部分**TODO**部分，我们也提供了额外的指导，请仔细阅读。

Part 1: Toy Model and Training Pipeline (50 pts)

NOTE: 该部分涉及的文件包括： `scripts/train.py`, `scripts/test.py`, `models/SimpleCNN.py`, `data/prepare_data.py`.

在这一部分，你将使用PyTorch实现一个简单的CNN模型，并完成模型在CIFAR-10数据集上的训练流程，同时使用Wandb记录实验过程。

具体而言，请完成以下几个子部分：

1. 在 `models/SimpleCNN.py` 中实现一个简易的CNN模型（`forward_map` 在该部分无需完成）。为保证模型的简易性，模型结构必须满足以下要求：1) 仅包含两个卷积层（以及对应的池化层）；2) 仅包含两个全连接层。
2. 在 `data/prepare_data.py` 中完成 `prepare_cifar10_datasets()` 函数，实现数据集的正确读取与划分。
3. 根据 `scripts/train.py` 中的具体指示，完成模型训练的流程。这包括：random seed的固定（保证可复现性，最终测试时我们将固定随机种子为0）、dataset与dataloader的构建、optimizer（以及optionally, scheduler）的构建、模型的前向传播/反向传播/优化、模型的正确保存，以及通过Wandb进行实验记录。

你需要通过Wandb记录以下内容： training loss (both step- and epoch-level), validation accuracy (epoch-level), best validation accuracy (epoch-level)。

4. 根据 `scripts/test.py` 中的具体指示，完成模型checkpoint的读取及模型在test dataset上的测试。

完成后，请你运行

```
python scripts/train.py --model SimpleCNN
```

来进行模型的训练。请确保你的模型超参数与优化后的模型参数保存在了正确的位置。此后请运行

```
python scripts/test.py --model SimpleCNN
```

来评估模型在测试集上的精度。

该部分的评分依据：

1. 代码的完成度（20 pts）。
2. Wandb对以上metric的记录情况（10 pts）。
3. 模型在测试集上的分类精度（20 pts）。

请注意：

1. 你可以通过调整模型超参数、实现任意训练数据增广、调整optimizer/scheduler、调整batch_size等方式来提升模型性能；但请勿调整total epoch number。
2. 为了保证模型的简易性，**请确保模型总参数量不超过100K**。在你的 `scripts/train.py` 中，我们通过 `assertion` 确保了这一点；在成绩评测时，我们也会首先检查这一点；如果超过要求的参数量，将扣除部分分数。

Part 2: Simple Vision Transformer (50 pts)

NOTE: 该部分涉及的文件包括： `models/SimpleViT.py`, `models/SimpleCNN.py`, `models/SimpleComposition.py`.

在这一部分，你将使用PyTorch实现一个简易版本的[Vision Transformer](#) (ViT)，并在CIFAR-10数据集上进行测试。

ViT摒弃了CNN中的若干归纳偏置（如平移不变性），将图片视为一个序列数据，并利用Transformer进行处理。在大规模数据集和模型参数下，ViT实现了与CNN模型相匹配甚至更好的性能，并展现出Transformer的scaling law。

在这一部分，你需要首先：

1. 在 `models/SimpleViT.py` 中完成Vision Transformer的所有构建模块，包括：Multi-head attention、前馈神经网络、Transformer Encoder（由前两个模块组成）、以及ViT（由前三个模块组成）。

完成后，你可以简单地进行超参数调试，并运行

```
python scripts/train.py --model SimpleViT

python scripts/test.py --model SimpleViT
```

来完成简单的训练和测试（该模型测试精度不纳入评分，因此无需深入调参；但请通过Wandb记录其相关指标，以供后续比较）。

你可能会发现，在相似的甚至于更高的模型参数量下，Simple ViT的表现一般，效果甚至有时不如Simple CNN。事实上，这是预期的结果，因为ViT摒弃了CNN的归纳偏置，因此在CIFAR-10这种数据规模较小、模型参数较少的情况下学习效果较差。然而，通过将二者结合，可以缓解这一问题：在原Simple ViT中，输入为原始图片，并在其上进行patch划分；而我们可以将原始图片替换为由CNN提取出的深层feature map，并在其上应用ViT，从而结合二者的优点。

请你继续完成以下内容：

2. 完成 `models/SimpleCNN.py` 中的 `forward_map` 函数，以实现CNN的feature map输出。
3. 完成 `models/SimpleComposition.py` 中的SimpleComposition模型构建。

完成后，你可以运行

```
python scripts/train.py --model SimpleComposition

python scripts/test.py --model SimpleComposition
```

来完成模型的训练和测试（该模型测试精度纳入评判标准，请你进行必要的调参）。

该部分的评分依据：

1. SimpleViT代码的完成度（20 pts）。
2. SimpleComposition（及SimpleCNN中forward_map部分）代码的完成度（10 pts）。
3. 通过Wandb对以上三个模型（i.e., SimpleCNN, SimpleViT, SimpleComposition）的第一部分中要求的指标进行记录，并且将三个模型的指标展示在同一张图上进行对比分析。（5pts）
4. SimpleComposition模型在测试集上的分类精度（15 pts）。

请注意：

1. 你应当使用PyTorch来完成这一部分，但不允许调用PyTorch内置的Transformer、TransformerEncoder模块：请确保完全由自己搭建Transformer。我们将在最终测试环境中通过限制可调用模块来保证这一点。
2. 对于SimpleComposition，可调整参数的限制同第一部分；该部分对模型参数量限制为120K。