南京農業大學

神经网络与深度学习

实验：面向土壤多项指标预测的神经网络建模与分析

姓　　名：叶俊泽

班　　级：人工智能 221

专　　业：人工智能

学　　号：11522105

2024 年 12 月

实验完成情况自查表

任务	完成情况	说明
第三方计算平台的使用	是 ☐ 否 ✓	在本地的一台拥有 4060 Ti 的 GPU 上训练模型
深度学习框架的选择	是 ✓ 否 ☐	CUDA 11.8 + Pytoch 2.1.5
数据集加载	是 ✓ 否 ☐	完成
数据集检视	是 ✓ 否 ☐	完成
数据预处理	是 ✓ 否 ☐	完成
模型搭建/调用	是 ✓ 否 ☐	完成
模型训练	是 ✓ 否 ☐	完成
模型测试	是 ✓ 否 ☐	完成

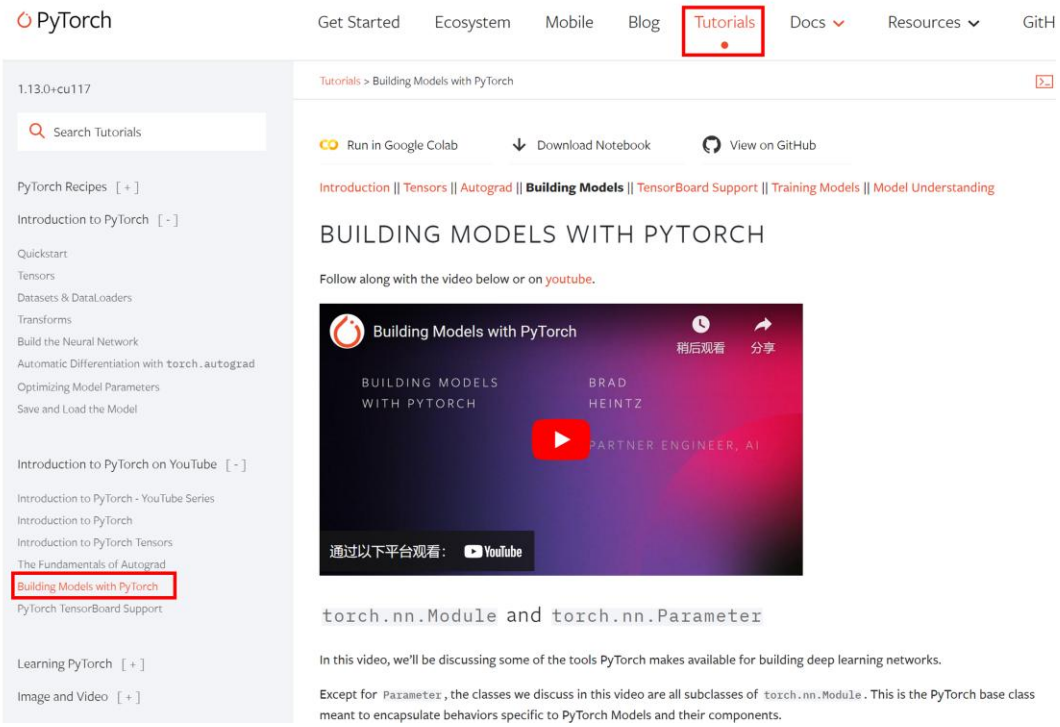
一、实验目的

掌握任意一种深度学习框架（Tensorflow、PyTorch、MindSpore、PaddlePaddle 等）的使用，部署自定义卷积神经网络（Convolutional Neural Network, CNN），完成多项土壤指标预测任务（回归任务）。

二、实验原理

参考教材 P115-121。

同时，可参考各深度学习框架官网教程（Tutorials），以 PyTorch 为例，如下图所示。



同时，各深度学习框架提供了预训练（Pre-trained）经典卷积神经网络库，例如 TensorFlow Hub（https://www.tensorflow.org/hub?hl=zh-cn）、PyTorch Hub（https://pytorch.org/hub/）、MindSpore ModelZoo（https://gitee.com/mindspore/models）等，调用方式请自行访问相关网页查阅。

三、实验内容

1. 使用未经预处理的完整 Record 作为输入（一个 Record 包含 4200 data points），分别构建并训练 ResNet18、ResNet34、ResNet50 模型，目标输出为 8 个土壤指标（pH in $CaCl_2$、pH in H_2O、OC、$CaCO_3$、N、P、K 和 CEC），对比不同深度神经网络的预测精度（使用 R^2 和 RMSE 作为评价指标）。

2. 参考论文 2.3 章节，使用 Abs-SG0、Abs-SG0-SNV、Abs-SG1、Abs-SG1-SNV、Abs-SG2 和 Abs-SG2-SNV 六种光谱预处理方法，实现原始光谱的变换，分别测试不同光谱预处理方法对模型预测性能的影响。

3. 参考论文 2.3 章节，移除 400-499.5nm 和 2450-2499.5nm 的波段，使用降采样的方式降低数据维度，分别合并 5nm、10nm 和 15nm 范围内的波长，即得到 390、195、130 个波长，将其作为网络的输入训练模型，对比不同下采样频率对模型预测性能的影响。

4. 参考论文 2.4 章节，使用 SHAP 值方法提取特征光谱，依据 SHAP 值分别得到对应排名前 10 的特征光谱波段。

5. 验证 1×1 卷积核的作用。在网络中添加 1×1 卷积核，对比添加前后模型的预测性能。

四、实验步骤

本实验包含以下流程：

1. 加载并读取数据集；
2. 数据预处理（去噪、标准化、训练集/测试集划分等）；
3. 构建卷积神经网络；
4. 训练、测试网络。

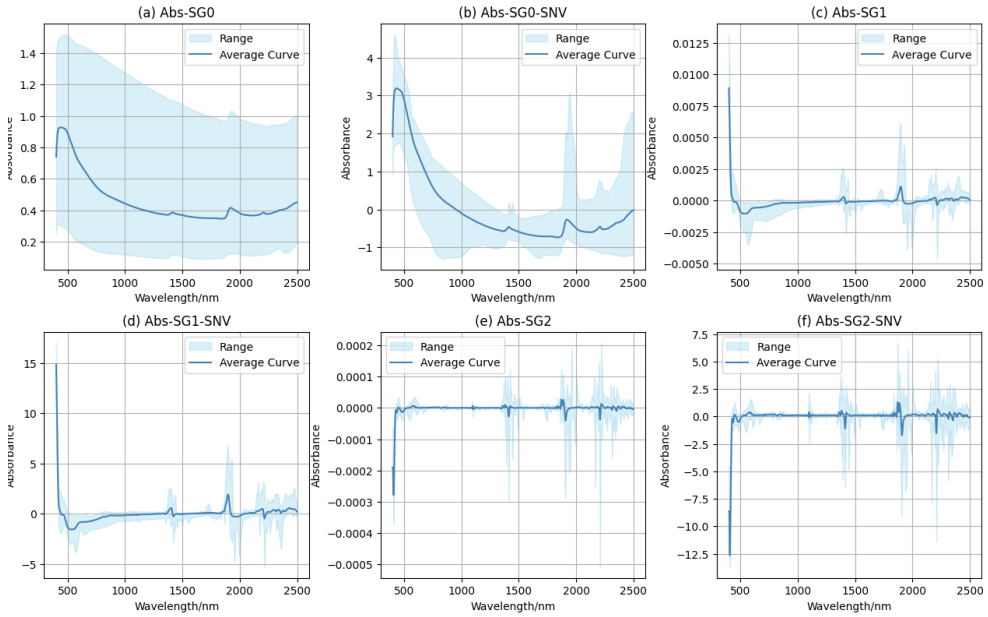
五、实验过程及结果展示

1. LUCAS Soil Dataset 中共提供了 19035 条数据记录（Records），请按照 8：2 的比例将原始数据集划分为训练集与验证集，并对数据集进行数理统计分析，填写下表。

表 1 数据集划分与分布统计

Soil properties	Training					Testing				
	Samples	Min	Max	Mean	Standard deviation	Samples	Min	Max	Mean	Standard deviation
OC	15228	0.00	586.80	50.38	91.97	3807	0.00	577.00	48.39	88.40
N	15228	0.00	38.60	2.95	3.80	3807	0.00	34.40	2.83	3.55
P	15228	0.00	789.80	30.12	31.76	3807	0.00	1366.40	29.87	36.90
K	15228	0.00	7342.00	198.27	235.55	3807	0.00	4665.00	192.18	202.28
CEC	15228	0.00	234.00	15.88	14.81	3807	0.00	146.00	15.24	13.08
pH.in. $CaCl_2$	15228	2.61	9.25	5.60	1.43	3807	2.66	8.07	5.58	1.43
pH.in.in H_2O	15228	3.21	10.08	6.20	1.35	3807	3.50	9.65	6.19	1.35
$CaCO_3$	15228	0.00	944.00	51.86	126.10	3807	0.00	882.00	50.59	122.13

2. 使用 Abs-SG0、Abs-SG0-SNV、Abs-SG1、Abs-SG1-SNV、Abs-SG2 和 Abs-SG2-SNV 六种光谱预处理方法，实现原始光谱的变换，并绘制图片展示。



3. 参考论文 Figure 1(e)给出的参数，使用任一深度学习框架自定义或调用 ResNet18、ResNet34、ResNet50 模型，分别记为模型 A、B、C。展示构建/调用模型的代码截图，并使用相关 API 打印模型 A、B、C 参数。例如 Tensorflow 的 model.summary()、PyTorch 的 torchsummary 等。

（构建/调用神经网络的代码截图）

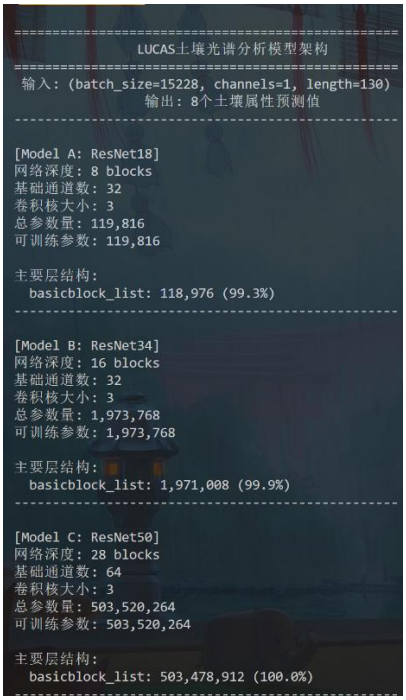
```python
class ResNet1D_MultiTask(resnet1d.ResNet1D):
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)

        # 安全地获取特征维度
        if not hasattr(self, 'dense'):
            raise AttributeError("Base model doesn't have 'dense' layer")

        in_features = self.dense.in_features

        # 移除原始的预测层
        delattr(self, 'dense')

        # 添加多任务预测头
        self.prediction_head = nn.Sequential(
            # 第一层：512 -> 256
            nn.Linear(in_features, in_features//2),
            nn.BatchNorm1d(in_features//2),
            nn.ReLU(),
            nn.Dropout(p=0.3),
            # 第二层：256 -> 128
            nn.Linear(in_features//2, in_features//4),
            nn.BatchNorm1d(in_features//4),
            nn.ReLU(),
            nn.Dropout(p=0.3),
            # 输出层：128 -> 8
            nn.Linear(in_features//4, 8)
        )
```

```python
def forward(self, x):
    # 获取特征提取器的输出
    out = x

    # first conv
    out = self.first_block_conv(out)
    if self.use_bn:
        out = self.first_block_bn(out)
    out = self.first_block_relu(out)

    # residual blocks
    for i_block in range(self.n_block):
        net = self.basicblock_list[i_block]
        out = net(out)

    # 特征聚合
    if self.use_bn:
        out = self.final_bn(out)
    out = self.final_relu(out)
    out = out.mean(-1)  # global average pooling

    # 通过预测头
    out = self.prediction_head(out)

    return out  # 现在直接输出8个指标的预测值
```

```python
def get_model(model_type):
    if model_type == 'A':  # ResNet18
        return ResNet1D_MultiTask(
            in_channels=1,
            base_filters=32,  # 减小base_filters，降低显存占用
            kernel_size=3,  # 使用3x3卷积核
            stride=2,
            groups=1,
            n_block=8,  # ResNet18的配置
            n_classes=8
        )
    elif model_type == 'B':  # ResNet34
        return ResNet1D_MultiTask(
            in_channels=1,
            base_filters=32,  # 调整base_filters
            kernel_size=3,  # 使用3x3卷积核
            stride=2,
            groups=1,
            n_block=16,  # ResNet34的配置
            n_classes=8
        )
    elif model_type == 'C':  # ResNet50
        return ResNet1D_MultiTask(
            in_channels=1,
            base_filters=32,  # 调整base_filters
            kernel_size=3,  # 使用3x3卷积核
            stride=2,
            groups=1,
            n_block=24,  # ResNet50的配置
            n_classes=8
        )
    else:
        raise ValueError("Invalid model type. Choose 'A' for ResNet
```

```python
# 模型参数设置
model = get_model('C')

# 损失函数
criterion = nn.SmoothL1Loss()

# 优化器
optimizer = torch.optim.AdamW(model.parameters(), lr=1e-3, weight_decay=1e-5)
scheduler = torch.optim.lr_scheduler.StepLR(optimizer, step_size=10, gamma=0.81)
```

（打印模型结构结果，示例如下，在提交的实验报告中需删除下图）



4. 说明本实验中使用的优化器与损失函数，并展示相关代码截图。

（文字说明）

优化器

- **AdamW**：这是 Adam 优化器的一个变体，包含权重衰减（L2 正则化）以提高泛化能力。学习率设置为 **1e-3**，权重衰减为 **1e-5**。

损失函数

- **SmoothL1Loss**：也称为 Huber 损失，此损失函数比均方误差损失对异常值不太敏感。通常用于需要对异常值具有鲁棒性的回归任务。

学习率调度器

- **StepLR**：该调度器通过因子（**gamma=0.81**）每 **10** 个 epoch 调整一次优化器的学习率，使学习率随时间逐渐衰减。

（代码截图）

```
# 模型参数设置
model = get_model('C')

# 损失函数
criterion = nn.SmoothL1Loss()

# 优化器
optimizer = torch.optim.AdamW(model.parameters(), lr=1e-3, weight_decay=1e-5)
scheduler = torch.optim.lr_scheduler.StepLR(optimizer, step_size=10, gamma=0.81)
```

5. 说明本实验采用的超参数设置，并展示相关代码截图。

（文字说明）

模型配置

- **模型类型：**

    - 'A': ResNet18
    - 'B': ResNet34
    - 'C': ResNet50
- **通用参数：**
    - in_channels=1：输入通道数，适用于单通道输入（如单维时间序列数据）。
    - base_filters=32：基础滤波器数量，用于控制网络复杂度和特征提取能力。
    - kernel_size=3：卷积核大小，为每层卷积操作定义感受野范围。
    - stride=2：卷积步长，决定特征图的下采样速率。
    - groups=1：卷积组数，设置为 1 表示标准卷积操作。
    - n_classes=8：输出类别数，模型最终预测的分类数目。
- **块数（n_block）：**
    - 8 对应 ResNet18
    - 16 对应 ResNet34
    - 24 对应 ResNet50

ResNet1D_MultiTask 类的预测头设计

总体结构说明

预测头是模型的最后部分，用于处理高维特征并生成最终输出结果。通过分层设计，逐步降低特征维度，同时加入正则化策略（批归一化和 Dropout）以增强模型的稳定性和泛化能力。预测头的最终输出适配多任务分类，输出类别为 8。

具体层次设计

1.**第一层**

- 使用全连接层将输入维度减少一半（in_features → in_features/2），实现特征降维。
- 引入**批归一化**（Batch Normalization）对中间特征进行归一化处理，减少梯度消失或爆炸的风险。
- 通过 **ReLU 激活函数**增加非线性建模能力，使模型能够学习复杂的非线性映射关系。
- 增加 **Dropout（p=0.3）**，随机失活部分神经元以降低过拟合风险。

2.**第二层**

- 进一步通过全连接层将特征维度从 in_features/2 降至 in_features/4，继续特征精简。
- 再次应用批归一化、ReLU 激活函数和 Dropout 操作，与第一层保持一致的正则化设计逻辑。

3.**输出层**

- 最终采用全连接层将降维后的特征映射到具体任务的输出空间，输出 8 个类别的预测结果。

训练配置

损失函数

- criterion = nn.SmoothL1Loss()：使用 SmoothL1 损失函数（Huber 损失）处理回归任务，具备对异常值的鲁棒性，有助于模型稳定优化。

优化器

- optimizer = torch.optim.AdamW(model.parameters(), lr=1e-3, weight_decay=1e-5)：采用 AdamW 优化器，结合权重衰减机制，提高模型的泛化能力。
    - **学习率**：1e-3
    - **权重衰减**：1e-5

学习率调度器

- scheduler = torch.optim.lr_scheduler.StepLR(optimizer, step_size=10, gamma=0.81)：学习率每隔 10 个 epoch 按 0.81 的比例衰减，以提高训练的收敛性和稳定性。
    - **步长**：10
    - **衰减率（gamma）**：0.81

数据加载

- **训练集加载器**：train_loader = DataLoader(train_dataset, batch_size=256, shuffle=True)
    - 批量大小：256
    - 随机打乱数据以提升训练效果。
- **测试集加载器**：test_loader = DataLoader(test_dataset, batch_size=256, shuffle=False)
    - 批量大小：256
    - 保持数据顺序以确保测试的稳定性和一致性。

（代码截图）

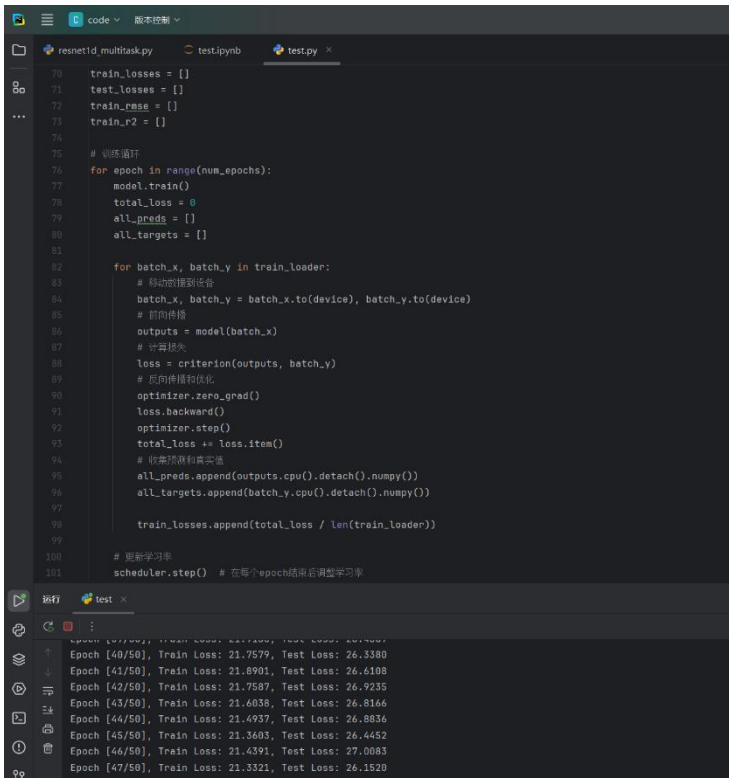
```python
def get_model(model_type):
    if model_type == 'A':  # ResNet18
        return ResNet1D_MultiTask(
            in_channels=1,
            base_filters=32,  # 减小base_filters，降低显存占用
            kernel_size=3,  # 使用3x3卷积核
            stride=2,
            groups=1,
            n_block=8,  # ResNet18的配置
            n_classes=8
        )
    elif model_type == 'B':  # ResNet34
        return ResNet1D_MultiTask(
            in_channels=1,
            base_filters=32,  # 调整base_filters
            kernel_size=3,  # 使用3x3卷积核
            stride=2,
            groups=1,
            n_block=16,  # ResNet34的配置
            n_classes=8
        )
    elif model_type == 'C':  # ResNet50
        return ResNet1D_MultiTask(
            in_channels=1,
            base_filters=32,  # 调整base_filters
            kernel_size=3,  # 使用3x3卷积核
            stride=2,
            groups=1,
            n_block=24,  # ResNet50的配置
            n_classes=8
        )
```

```python
class ResNet1D_MultiTask(resnet1d.ResNet1D):
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)

        # 安全地获取特征维度
        if not hasattr(self, 'dense'):
            raise AttributeError("Base model doesn't hav

        in_features = self.dense.in_features

        # 移除原始的预测层
        delattr(self, 'dense')

        # 添加多任务预测头
        self.prediction_head = nn.Sequential(
            # 第一层：512 -> 256
            nn.Linear(in_features, in_features//2),
            nn.BatchNorm1d(in_features//2),
            nn.ReLU(),
            nn.Dropout(p=0.3),
            # 第二层：256 -> 128
            nn.Linear(in_features//2, in_features//4),
            nn.BatchNorm1d(in_features//4),
            nn.ReLU(),
            nn.Dropout(p=0.3),
            # 输出层：128 -> 8
            nn.Linear(in_features//4, 8)
        )
```

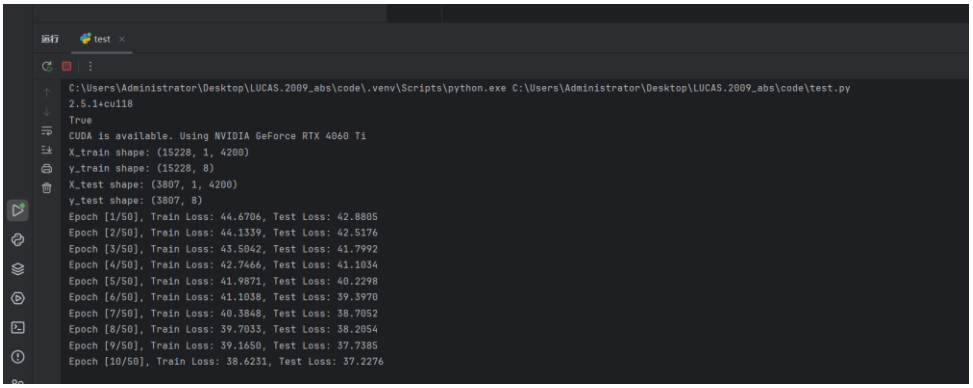
```python
# 模型参数设置
model = get_model('C')

# 损失函数
criterion = nn.SmoothL1Loss()

# 优化器
optimizer = torch.optim.AdamW(model.parameters(), lr=1e-3, weight_decay=1e-5)
scheduler = torch.optim.lr_scheduler.StepLR(optimizer, step_size=10, gamma=0.81)
```

6. 将未经预处理的原始数据作为输入，分别训练 A、B、C 模型，展示执行模型训练的代码截图，同时，请展示执行训练命令后，模型的训练过程，并根据迭代过程中 Training 和 Validation 的 Loss 绘制相应折线图。本实验采用 RMSE 和 R^2 作为模型性能的评价指标，请在训练过程中分别记录各指标的数值，并将数值填写在表格内，同时，参考论文图 3 绘制散点图。（**建议使用硬参数共享的多任务学习模型，若能力有限，可逐一训练模型预测各土壤指标，即训练 8 个 A 模型、8 个 B 模型、8 个 C 模型**）

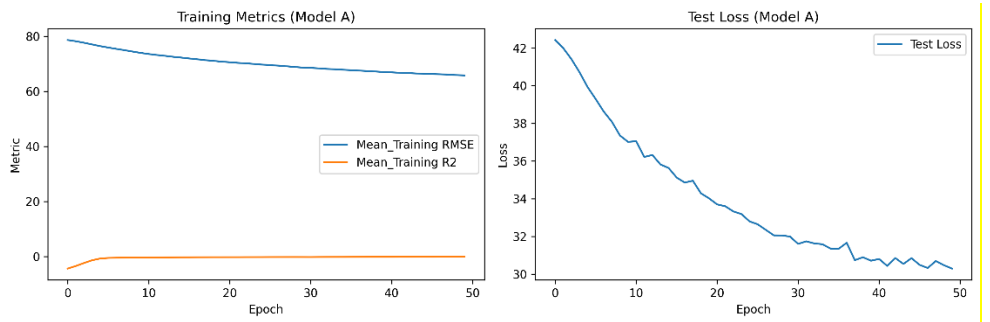
（执行训练的代码截图）

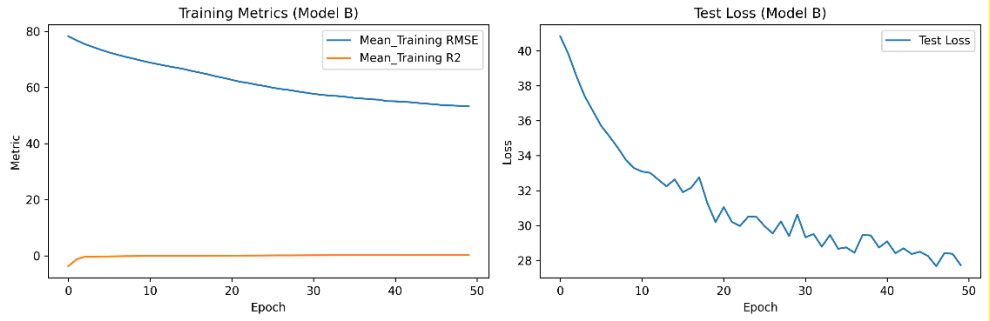


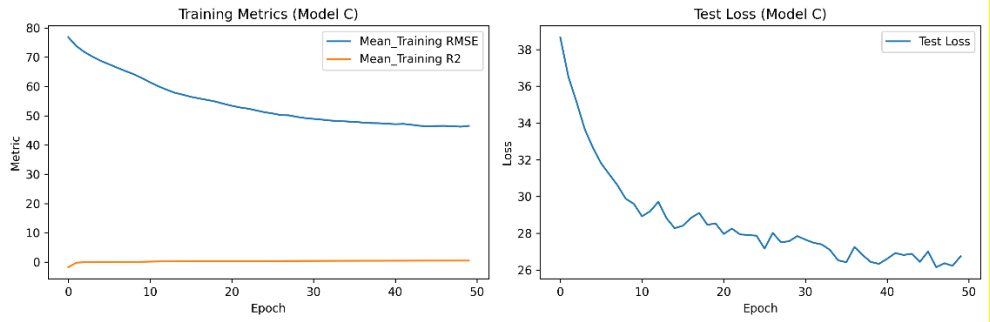
（训练过程输出结果图，训练过程示例结果如下图所示，在提交的实验报告中删除下图）



（折线图示例如下图所示，分别展示 A、B、C 模型的训练过程，即有 3 个图（或使用不同颜色的实线/虚线标注 3 个模型，只绘制 1 个图）。请在提交的实验报告中删除下图）

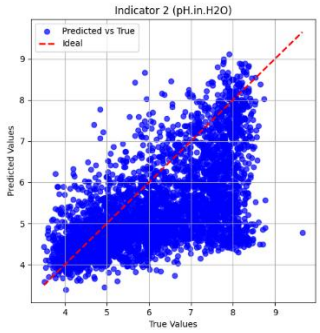


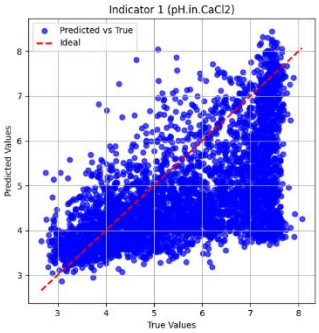


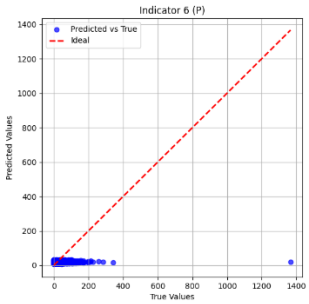


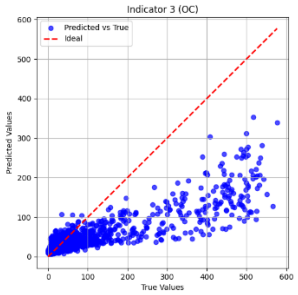
（将测试集的结果填写在以下表格内）

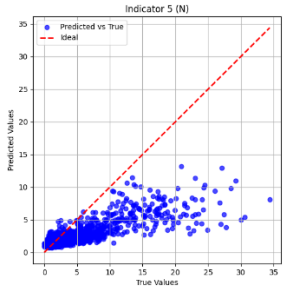
Soil properties	Evaluation indicators	模型 A	模型 B	模型 C
OC	RMSE	9.3670	8.2344	8.0283
	R^2	0.0148	0.4116	0.4683
N	RMSE	1.8129	1.6586	1.6682
	R^2	0.1423	0.3991	0.3850
P	RMSE	6.2296	6.1583	6.1934
	R^2	-0.1063	-0.0565	-0.0808
K	RMSE	15.0763	14.5700	14.2200
	R^2	-0.2626	-0.1014	0.0007
CEC	RMSE	3.6354	3.5120	3.4098
	R^2	-0.0210	0.1108	0.2098
pH in $CaCl_2$	RMSE	1.2717	1.2917	1.1903
	R^2	-0.2851	-0.3678	0.0136
pH in H_2O	RMSE	1.2939	1.2833	1.1664
	R^2	-0.5375	-0.4877	-0.0155
$CaCO_3$	RMSE	11.4724	10.9131	10.7531
	R^2	-0.1613	0.0491	0.1037

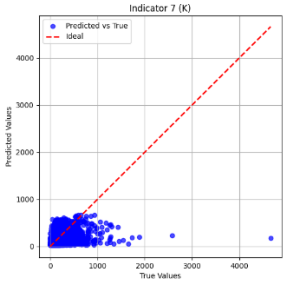


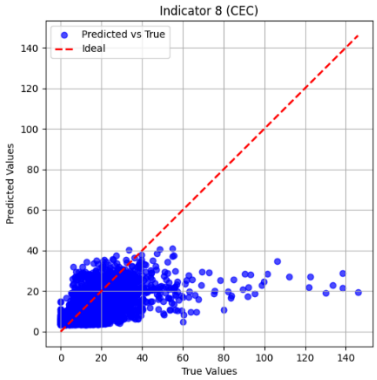


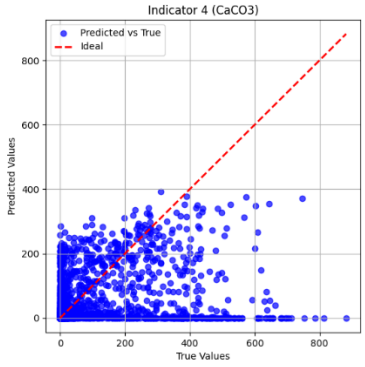












结论：综合考虑下，最优模型为 C。

7. 后续实验均采用步骤 6 中的最优模型，此时，将原始数据经六种光谱预处理方法后变换得到的数据作为输入，训练最优模型，请在训练过程中分别记录各指标的数值，并将数值填写在表格内。

（将测试集的结果填写在以下表格内）

Soil properties	Evaluation indicators	PP1	PP2	PP3	PP4	PP5	PP6
OC	RMSE	8.1022	6.0624	9.0927	6.9911	9.5270	5.8075
	R^2	0.4485	0.8271	0.1252	0.6943	-0.0543	0.8544
N	RMSE	1.6102	1.4266	1.8572	1.6105	1.9782	1.4128
	R^2	0.4662	0.6711	0.0553	0.4658	-0.2159	0.6836
P	RMSE	6.1459	6.1328	6.0900	5.9795	6.6043	5.8376
	R^2	-0.0480	-0.0392	-0.0105	0.0609	-0.3975	0.1469
K	RMSE	13.9761	13.8842	14.0982	13.8154	15.9916	12.9567
	R^2	0.0675	0.0918	0.0345	0.1097	-0.5983	0.3112

CEC	RMSE	3.2445	3.3248	3.6072	3.5810	4.1587	3.2131
	R^2	0.3522	0.2857	0.0103	0.0387	-0.7483	0.3770
pH in $CaCl_2$	RMSE	1.1329	1.0763	1.1943	0.9366	1.5963	0.8667
	R^2	0.1905	0.3406	0.0002	0.6220	-2.1907	0.7227
pH in H_2O	RMSE	1.1135	1.0515	1.1764	0.9237	1.5703	0.8609
	R^2	0.1565	0.3294	-0.0507	0.6007	-2.3361	0.6986
$CaCO_3$	RMSE	10.7757	8.5423	11.3249	7.3689	11.4900	6.3981
	R^2	0.0961	0.6430	-0.1028	0.8023	-0.1685	0.8877

注：PP 表示 Pre-processing。PP1: Abs-SG0、PP2: Abs-SG0-SNV、PP3: Abs-SG1、PP4: Abs-SG1-SNV、PP5: Abs-SG2 和 PP6: Abs-SG2-SNV。

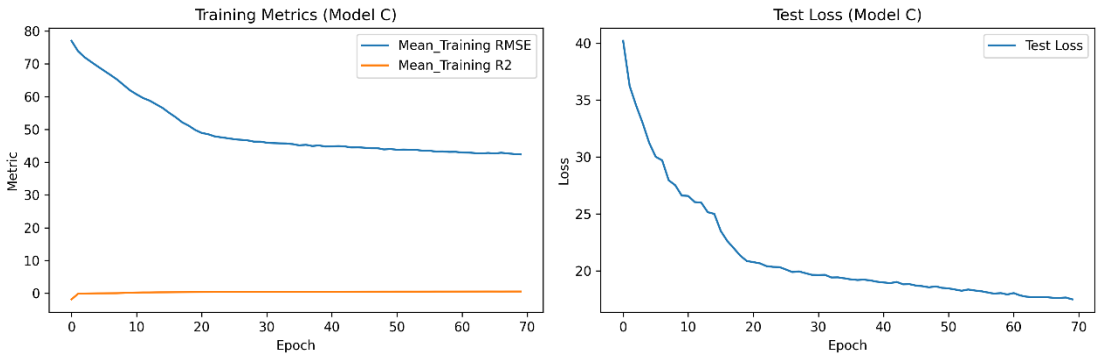
结论：综合考虑下，最优的光谱预处理方法为 PP6: Abs-SG2-SNV。

8. 后续实验均采用步骤 6 中的最优模型与步骤 7 中的最优光谱预处理方法，此时，将下采样后的数据作为输入，训练模型，请在训练过程中分别记录各指标的数值，并将数值填写在表格内。同时，请根据迭代过程中 Training 和 Validation 的 Loss 绘制相应折线图。

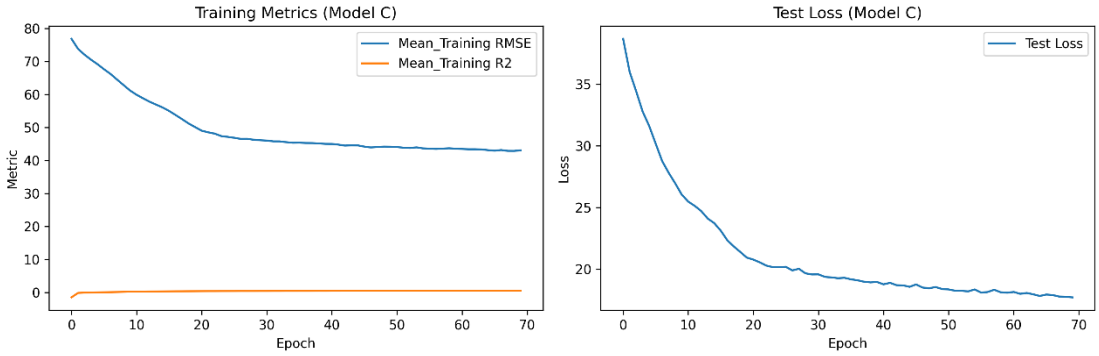
（将测试集的结果填写在以下表格内）

Soil properties	Evaluation indicators	1950 DPs	390 DPs	195 DPs	130 DPs
OC	RMSE	5.8075	5.6200	5.6382	5.2944
	R^2	0.8544	0.8723	0.8707	0.8994
N	RMSE	1.4128	1.3876	1.3601	1.2336
	R^2	0.6836	0.7056	0.7283	0.8161
P	RMSE	5.8376	5.9743	5.9834	5.8376
	R^2	0.1469	0.0642	0.0585	0.1469
K	RMSE	12.9567	13.3309	13.2234	12.5448
	R^2	0.3112	0.2281	0.2527	0.3947
CEC	RMSE	3.2131	3.1718	3.1393	2.8166
	R^2	0.3770	0.4084	0.4323	0.6321
pH in $CaCl_2$	RMSE	0.8667	0.9089	0.8906	0.8083
	R^2	0.7227	0.6646	0.6909	0.7902
pH in H_2O	RMSE	0.8609	0.8884	0.8649	0.7884
	R^2	0.6986	0.6582	0.6930	0.7881
$CaCO_3$	RMSE	6.3981	6.6892	6.3763	6.1148
	R^2	0.8877	0.8658	0.8892	0.9063

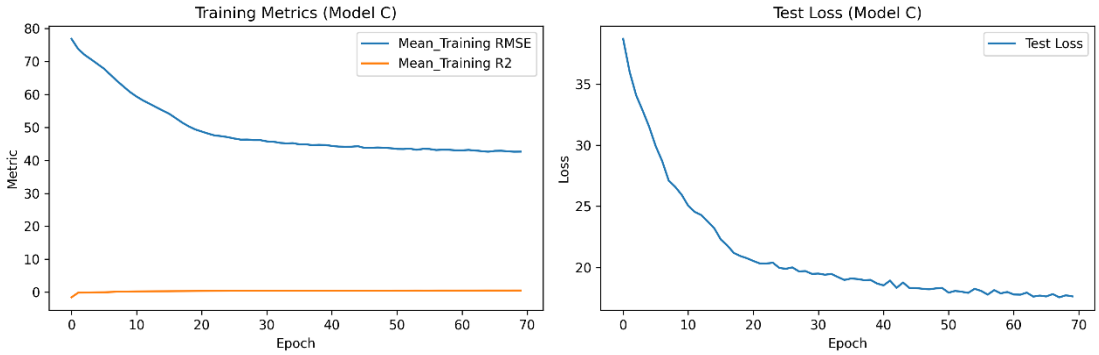
注：DPs 表示 Data points。1950 DPs 表示未经下采样的所有全部数据，390 DPs 表示经 5nm 下采样后得到的 390 个光谱波段，以此类推。



降采样窗口为 5



降采样窗口为 10

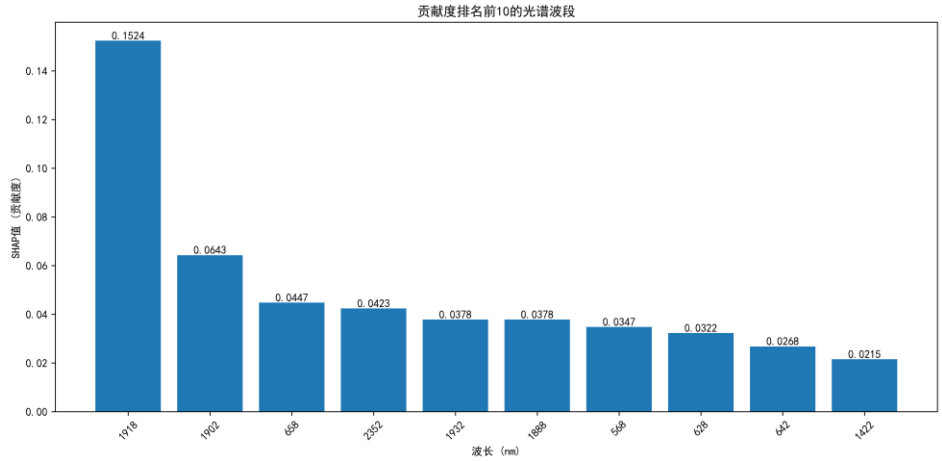


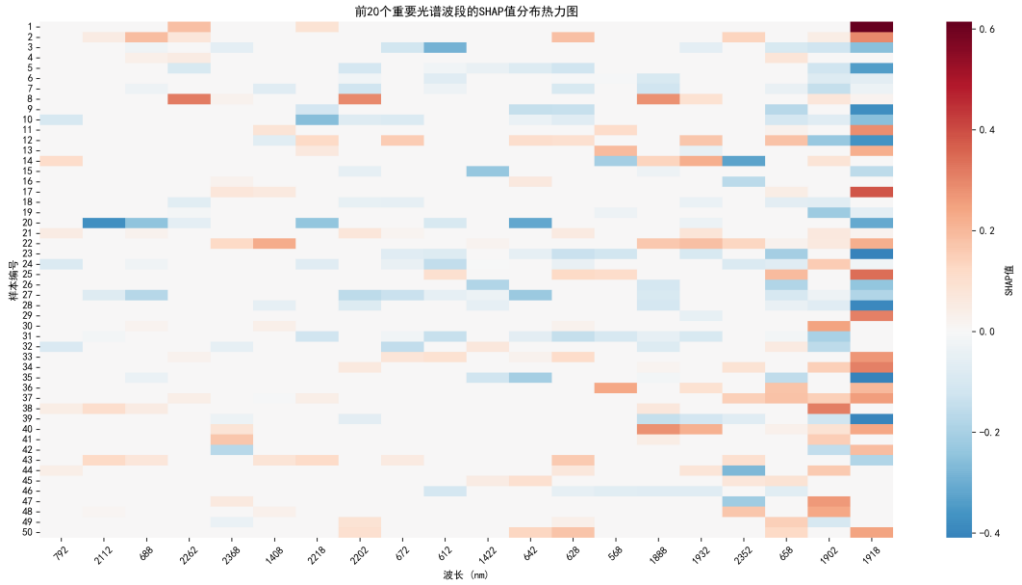
降采样窗口为 15

结论：最佳的下采样方式为 130DPS，得到 130 个光谱波段。

9. 后续实验均采用步骤 6 中的最优模型、步骤 7 中的最优光谱预处理方法、步骤 8 中的最优下采样方法的基础上，此时，使用 SHAP 值方法评价各光谱波段对预测结果的贡献率，筛选出贡献度排名前 10 的光谱波段，绘制蜂窝图与柱状图（可调用 SHAP 库实现：import shap）。

（蜂窝图与柱状图示例如下）





10. 在步骤 6（最优模型）、步骤 7（最优光谱预处理方法）、步骤 8（最优下采样方法）的基础上，在适当位置嵌入 1×1 卷积核，说明嵌入的位置及数量，再次训练模型，请在训练过程中分别记录各指标的数值，并将数值填写在表格内，同时，参考论文图 3 绘制散点图。

在最后的预测头部分，添加了两层的 1*1 卷积层。

- 第一层：

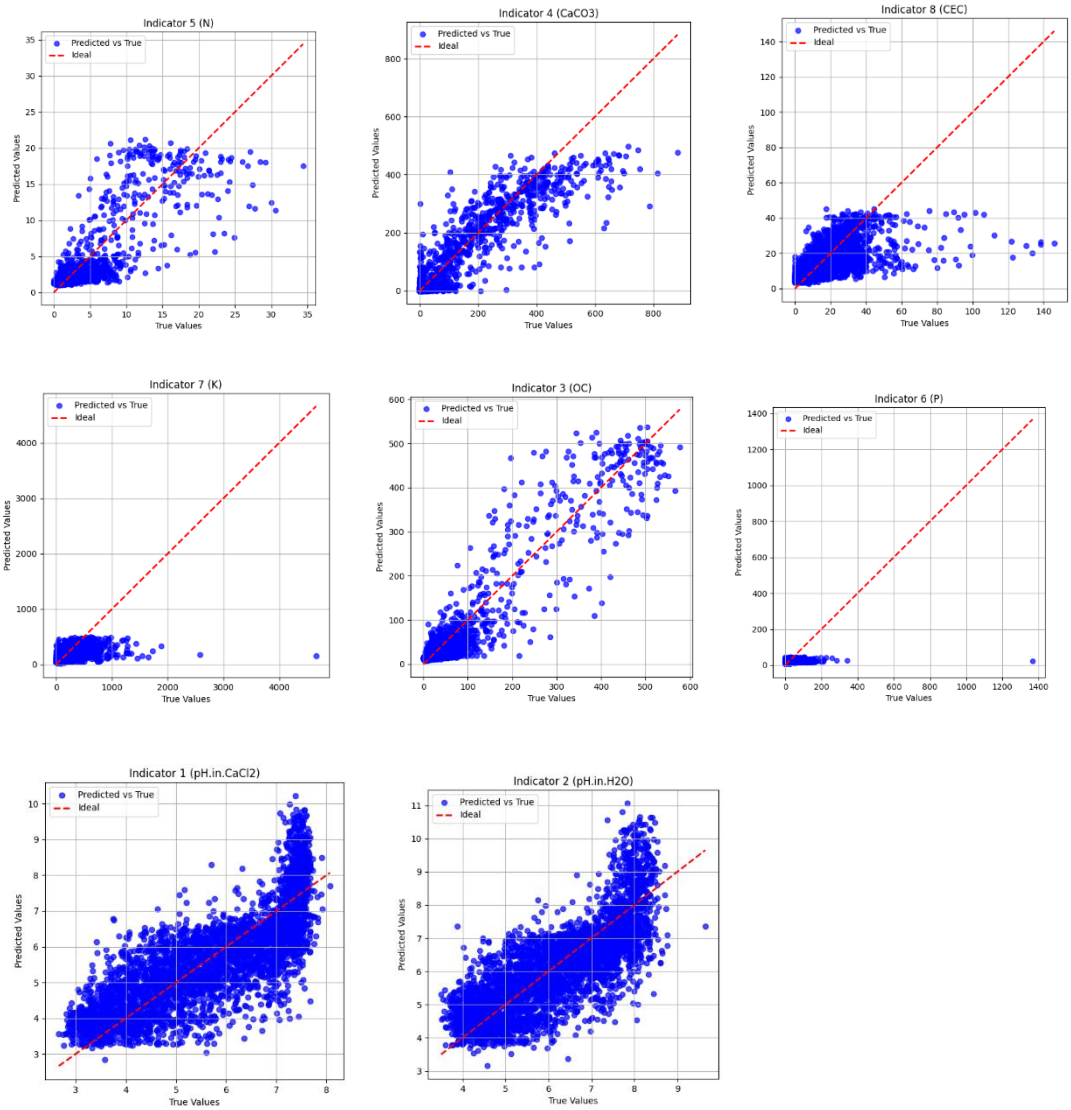
输入维度：in_features（512）；出维度：in_features // 2（256）；量：512×256=131,072

- 第二层：

输入维度：in_features // 2（256）；出维度：in_features // 4（128）；量：256×128=32,768

Soil properties	Evaluation indicators	With 1×1 Conv	Without 1×1 Conv
OC	RMSE	5.4777	5.2944

	R^2	0.8848	0.8994
N	RMSE	1.4176	1.2336
	R^2	0.6794	0.8161
P	RMSE	6.0383	5.8376
	R^2	0.0234	0.1469
K	RMSE	13.3519	12.5448
	R^2	0.2233	0.3947
CEC	RMSE	3.2411	2.8166
	R^2	0.3550	0.6321
pH in $CaCl_2$	RMSE	0.9515	0.8083
	R^2	0.5972	0.7902
pH in H_2O	RMSE	0.9461	0.7884
	R^2	0.5605	0.7881
$CaCO_3$	RMSE	6.7208	6.1148
	R^2	0.8632	0.9063



结论：综合考虑下，1×1 卷积核无助于提升模型的预测精度。

六、实验问题回答

1. 请简述优化器和损失函数的选择依据。

损失函数：nn.SmoothL1Loss()

- **1.**因为任务是回归问题，模型需要预测 8 个土壤指标，这些是连续值。SmoothL1Loss 兼顾 L1 和 L2 损失的优点，能够在处理数据中的异常值时保持对模型梯度的平稳优化。
- **2.** 1 维 ResNet50 模型具有深层网络结构，容易出现梯度爆炸或梯度消失的问题。SmoothL1Loss 平滑的梯度特性能更好地与深度模型结合，提高优化的稳定性。
- **3.**光谱数据通常包含噪声或异常值，这种损失函数的鲁棒性可有效降低异常值对训练过程的影响，提升模型对实际数据的适应能力。
-

优化器：torch.optim.AdamW

- **1.深度网络需求**：改进的 1 维 ResNet50 模型由于深度较深，参数量较大，对优化器的选择要求高。AdamW 结合了 Adam 的自适应学习率机制和权重衰减功能，非常适合深度网络。
    - **自适应学习率**：AdamW 会动态调整不同参数的学习率，适用于深度网络中参数分布不均的情况。
    - **权重衰减**：深度网络容易过拟合，尤其在光谱数据这样的高维输入下，权重衰减有助于限制网络的复杂度，增强模型的泛化能力。
- **2.训练效率**：在深层网络中，AdamW 优化器相比传统 SGD 优化器收敛速度更快，且对学习率的敏感性较低，更加稳定。

学习率调度器：torch.optim.lr_scheduler.StepLR

- **1.深度模型优化需求**：ResNet50 这样的深度模型在早期阶段需要较大的学习率快速收敛，而在后期则需要逐步降低学习率以稳定优化。StepLR 每 10 个 epoch 按比例 0.81 衰减学习率，符合深度网络的训练特性。
- **2.长时间训练的稳定性**：对深度网络来说，逐步衰减学习率可以防止在训练后期因较大的步长导致模型参数来回震荡，有助于模型接近最优解。

改进的 1 维 ResNet50 模型针对光谱数据预测土壤指标，选择 SmoothL1Loss 可以增强对异常值的鲁棒性，AdamW 优化器则提供了高效稳定的参数更新方式，同时结合 StepLR 逐步降低学习率，使深层网络在训练的各个阶段都能高效稳定地优化。这些选择与模型的深度、数据的高维性及任务的回归特性高度契合。

2. 解释过拟合和欠拟合现象产生的原因，并阐述解决这两个问题的可行方案。

过拟合现象

原因

1. **模型复杂度过高**：
    - 改进的 1 维 ResNet50 是一个深度网络，参数量较多，具有较强的拟合能力。如果训练数据不足，模型可能学习到训练数据中的噪声和异常特性，而无法很好地泛化到测试数据。
2. **数据样本有限**：
    - 如果光谱数据中训练样本的数量相对于模型的参数过少，模型容易记住训练集的特性，而非学习通用模式。

3. **高维输入数据：**
    - 如果事先没有通过降采样（bin_sizes[2]）降低了光谱的维度，光谱数据本身是高维的（4200），高维空间中的噪声可能影响模型的训练。
4. **训练时间过长：**
    - 如果训练的 epoch 数过多，模型可能过度拟合训练集，损失函数在训练集上持续减小，但测试集表现变差。

解决方案

1. **正则化：**
    - 已经采用了 AdamW 优化器的权重衰减（weight_decay=1e-5），可以进一步尝试增大权重衰减值以限制模型的复杂度。
    - 增加 **Dropout 层**：在 ResNet50 中的一些全连接层中添加 Dropout，例如 p=0.3，可以随机丢弃一些神经元，防止模型过拟合。
2. **数据下采样**：通过降采样（bin_sizes[2]）降低了光谱的维度 从 4200 到 130
3. **早停：**
    - 监控验证集的损失，在损失不再改善时停止训练，避免模型过度拟合。
4. **增大数据集：**
    - 收集更多的光谱数据，或者通过数据增强生成新的样本。

欠拟合现象

原因

1. **模型复杂度不足：**
    - 如果改进的 1 维 ResNet50 模型在改进过程中减少了太多的参数或层数，可能导致模型表达能力不足，无法捕获光谱数据中的复杂模式。
2. **数据预处理影响信息保留：**
    - 降采样（bin_sizes[2]）可能丢失了光谱数据中的关键特征，导致模型无法学习到有效的模式。
    - 数据预处理方法（如 Abs-SG1-SNV）可能对特定土壤指标不敏感，信息丢失后模型难以有效预测。
3. **优化不足：**
    - 虽然 AdamW 优化器有良好的表现，但学习率（lr=1e-3）可能过低，导致模型无法充分学习；也可能过高，导致在早期训练阶段模型未能找到较好的方向。

解决方案

1. **提升模型复杂度：**
    - 如果模型参数量过小，可以尝试增加 ResNet50 中的层数或通道数，提升模型的表达能力。
    - 引入注意力机制（例如 SE 模块或 CBAM），增强模型对光谱数据的关键特征提取能力。
2. **优化数据预处理：**
    - 尝试更适合土壤数据的预处理方法。例如，不同目标指标可能需要不同的预处理方法，针对具体指标进行方法优化。
    - 降采样时尝试更小的窗口大小（如 bin_sizes[1]），保留更多的光谱细节信息。
3. **调整超参数：**

- 调整学习率：尝试使用学习率搜索方法（如 Cyclical Learning Rates），找到适合的初始学习率。
- 适当增大学习率（如 lr=5e-3），加速模型的初期学习，避免欠拟合。

4. **增加训练时间：**
   - 如果训练时间不足（epoch 数量较少），可以适当增加训练 epoch 数，但需配合早停机制。

3. 关于 CNN 的可解释性研究是当前的热门，请阅读以下文献，结合实验设计，谈谈你对可解释性人工智能（Explainable Artificial Intelligence, XAI）的理解与认识。（可结合使用 SHAP 值提取对预测结果具有较高贡献度的特征光谱波段说明）

在深度学习的研究中，可解释性（Explainable Artificial Intelligence, XAI）已成为解决“黑箱问题”的重要方向，特别是对卷积神经网络（CNN）等复杂模型的分析，有助于提升模型的透明性和信任度。

XAI 研究现状

深度学习模型在图像、语音、文本等任务中的广泛应用，模型的复杂度显著提升，模型的“黑箱”特性日益突出。这种黑箱特性使得模型的预测结果虽然精准，但其决策依据难以直观理解，尤其在医学、金融等高风险领域引发了对可解释性的强烈需求。在农业领域，土壤指标预测任务同样对模型的透明性有较高要求。通过深度学习模型预测土壤中的关键成分（如有机质、氮、磷等），需要验证预测的科学性和合理性，而不是单纯依赖结果。因此，实验中对模型的输出结果进行解释，尤其是通过特征重要性分析找到对预测影响最大的光谱波段，成为必不可少的环节。

文献进一步从深度学习的角度分析了可解释性研究的现状，提出了包括可视化分析、敏感性分析和鲁棒性扰动测试等方法。其中，可视化分析通过提取中间层特征图或卷积核激活模式，可以直观展示模型“学到”的信息。

本次实验的应用

在实验设计中，SHAP（SHapley Additive exPlanations）是一个适用于深度学习模型的强大工具。SHAP 基于博弈论思想，可以量化每个输入特征对模型预测的贡献度。这种特征重要性分析可以直接应用于 4200 维的光谱数据上，识别出对 8 个土壤指标具有较大影响的关键波段。这种分析有多重好处：首先，它能够帮助科学家优化模型输入，降低数据维度，提高计算效率；其次，它可以为农业生产提供科学依据，比如强调某些光谱波段对预测的显著影响，指导传感器或遥感设备的研发。

在实际应用中，文献强调了可解释性在农业、医学和金融领域的重要意义。在土壤多指标预测中，除了提升模型性能外，解释性研究还可以指导农业实践。例如，通过模型分析发现某些光谱波段对土壤有机质预测的贡献度较高，那么这些波段的信息可能与土壤的物理或化学特性相关。这样的发现可以帮助农民制定更精准的施肥策略，也能推动传感设备的升级，使其聚焦于高影响波段，从而降低成本并提高效率。

模型的改进与期望

模型的可解释性不仅限于结果的理解，更涉及对模型内部结构和机制的剖析。例如，基于激活值最大化的技术可以帮助研究者理解哪些输入模式会激活神经网络的某些层或节点。在实验中，这可以通过对 ResNet50 中卷积层的激活值进行分析实现。如果特定光谱波段的变化使某层的神经元激活显著增强，说明该波段可能包含对特定土壤指标的重要信息。结合 SHAP 值分析，我们能够验证这种重要性是否一致，从而为解释模型决策提供多层次的支持。

从未来发展的角度看，本实验的研究可以进一步拓展到构建更具解释性的模型结构。例

如，在 ResNet50 的基础上引入注意力机制，让模型能够自适应地聚焦于关键波段，这种机制不仅可以提升模型的预测性能，还可以为解释提供更自然的依据。此外，CapsNet 等结构化网络也可以作为研究方向，其内置的空间关系建模能力可能更适合光谱数据的分析。

文献阅读：

[1] 陈珂锐，孟小峰. 机器学习的可解释性[J]. 计算机研究与发展，2020，57（9）：1971-1986.

[2] 成科扬，王宁，师文喜，詹永照. 深度学习可解释性研究进展[J]. 计算机研究与发展，2020，57（6）：1208-1217.

4. 在实验过程中，是否有遇到其他问题？

1. 最大的问题出现在模型的构建上。因为 Pytorch 的包的 ResNet 模型的卷积核以及池化是 2 维，但是本实验中的输入数据是 4200 维的列向量，不是处理图像，这里就需要对模型的残差模块进行改进。

但是，我并没有注意到对残差块之间的维度的改进，只是对卷积核、池化模块简单的改进，导致了输入的数据的维度和参数矩阵不匹配，模型的通道数也不匹配。在查阅了有关 ResNet1d 的有关文献后，成功对模型进行了改进。

2. 第二问题出现在模型的超参数的设定中，刚开始我把模型的通道数设置的过高（128），同时，设置的卷积核也过大（1*7）。导致机器训练过慢，而且效果也不理想。之后我适当减少了通道数为 32 以及使用 3 大小的卷积核，机器在一定时间内的效果表现良好。

3.采用不同的降采样处理数据的时候，会导致输入的通道改变，为了解决这个问题，卷积层和全连接层的设计并没有对固定输入维度做出强约束。特别是卷积层使用了 My-onv1dSame ，支持输入长度的变化，同时 global average pooling (即 out.mean(-1)) 负责对最后一维的数据进行聚合，使输出的特征维度不依赖输入序列的长度。这种设计使得即使输入长度变化，模型仍能正常处理。

同样的，BasicBlock 和网络主体中，残差连接部分只需保证输入和输出的通道数对齐，因此数据在降采样过程中长度改变后，残差块仍然可以正常工作。这一灵活性也得益于代码中 F.pad 的使用，它动态补全了维度。

七、实验心得体会

在此次神经网络建模实验中，深刻感受到深度学习框架在处理高维数据上的强大能力，同时也领悟到细致设计和参数优化的重要性。这次实验围绕土壤多项指标的预测任务展开，涉及数据预处理、模型构建与训练、性能分析等多个环节，实验内容紧凑而具有挑战性。

对于数据的处理认识到，科学合理的预处理手段对模型性能至关重要。通过对比六种光谱预处理方法，我清楚地看到不同方法对预测精度的影响。这不仅帮助我了解了 Abs-SG-SNV 变换在光谱特征提取中的优势，也让我明白数据噪声去除和特征保留间的平衡尤为重要。同时，降采样操作虽然降低了数据维度，但它也可能导致部分信息丢失，因此选取适当的采样率显得尤为关键。在实验中，我最终选择 130 个波长的下采样方法，兼顾了计算效率和预测精度，

在模型构建方面，本次实验让我深入体会到改进经典模型的必要性与挑战性。由于 1 维 ResNet 模型需要适配光谱数据的列向量输入，对残差模块进行了 1D 化改进，并尝试加入 1×1 卷积核 来降低参数复杂度。然而，实验结果显示，1×1 卷积核的引入对模型的预测性能并没有显著提升，甚至在某些配置下略微降低了精度。这让我意识到，模型复杂度的增加并不总是带来性能的提升，合理的结构设计必须充分考虑任务特点和数据模式。这次尝试尽管未能如愿取得优化效果，但它帮助我加深了对深度学习模型内部逻辑和特定操作作用的理解。

实验中也暴露了一些问题。初期，我在模型通道数和卷积核大小的设置上出现了失误，过大的参数导致训练效率低下，并引发过拟合现象。在后续优化中，我逐渐调小参数，同时采用 AdamW 优化器结合 StepLR 学习率调度器，这些调整显著提升了模型的收敛速度和性能稳定性。这个过程让我认识到，优化器和损失函数的选择并非单纯的技术性操作，而是与任务和数据特点紧密相关的设计决策。

此外，本次实验让我对可解释性人工智能（Explainable AI）的重要性有了更深刻的理解。在使用 SHAP 值分析特征光谱时，我发现部分光谱波段对模型预测具有较大贡献，而另一些波段几乎没有显著作用。这种分析不仅让我对模型的决策逻辑有了更加直观的认识，还为后续的模型优化提供了方向。在实际应用中，能解释模型的预测结果对于提高用户信任和指导实验设计尤为关键。

这次实验虽然充满挑战，但帮助我掌握了 1 维深度神经网络的建模与优化方法，同时让我在实际问题解决中提升了编程能力、理论知识与问题分析能力。尽管部分尝试未达到预期效果，但理论与实践的结合让我更加明确未来努力的方向，也让我对深度学习在解决复杂实际问题中的潜力充满信心。