最优学习率搜索（一）：基于学习率扫描的最优学习率搜索

朱梦

初稿于 2025 年 6 月 29 日，修改于 2025-08-15

1. 问题定义及其分析

对于深层神经网络模型，学习率过大容易导致训练震荡甚至发散，过小则收敛缓慢浪费算力资源。因此，选取一个合适的学习率对模型的训练是至关重要的。基于批处理大小和学习率的缩放规律、模型宽度和学习率的缩放规律、模型深度和学习率的缩放规律，因此可以在某个较小的批处理大小且较合适的模型尺寸下对学习率进行搜索，搜出一个最优的学习率，然后基于缩放规律进行尺度迁移。

2. 问题解决

预期目标：在某个较小的批处理大小且较合适的模型尺寸下对学习率进行搜索，搜出一个最优的学习率。

```python
def lr_finder(
    model: nn.Module,
    OptType: Type[torch.optim.Optimizer],
    train_dataloader: DataLoader,
    val_dataloader: DataLoader,
    Criterion: Type[_Loss] = nn.CrossEntropyLoss,
    start_lr: float = 1e-8,
    end_lr: float = 0.1,
    warmup_epochs: int = 0,
    training_epochs: int = 1,
) -> Dict[float, float]:
    assert warmup_epochs < training_epochs

    rsts = {}
    lr_range = np.logspace(np.log10(start_lr), np.log10(end_lr), 10)
    init_wts = deepcopy(model.state_dict())
    device = next(model.parameters()).device
    criterion = Criterion().to(device)

    print("\n===== 学习率快速扫描 =====")
```

```
    for i, lr in enumerate(lr_range):
        if i != 0:
            model.load_state_dict(init_wts)

        warmup_batches = warmup_epochs * len(train_dataloader)
        batch_cnt = 1
        if warmup_batches != 0:
            optimizer = OptType(model.parameters(), lr=lr * batch_cnt /
                warmup_batches)   # 学习率预热
        else:
            optimizer = OptType(model.parameters(), lr=lr)

        model.train()
        for _ in range(training_epochs):
            for inputs, targets in train_dataloader:
                inputs, targets = inputs.to(device), targets.to(device)
                optimizer.zero_grad()
                outputs = model(inputs)
                loss = criterion(outputs, targets)
                loss.backward()
                optimizer.step()

                batch_cnt += 1
                if batch_cnt < warmup_batches:
                    for param_group in optimizer.param_groups:
                        param_group['lr'] = lr * batch_cnt / warmup_batches

        model.eval()
        correct, total = 0, 0
        with torch.no_grad():
            for inputs, targets in val_dataloader:
                inputs, targets = inputs.to(device), targets.to(device)
                outputs = model(inputs)
                _, preds = torch.max(outputs, 1)
                correct += (preds == targets).sum().item()
                total += targets.size(0)

        acc = correct / total
        rsts[lr] = acc
        print(f"LR={lr:.2e}: 准确率={acc:.4f}")

    return rsts
```

通过返回的结果，既可以求解出最优学习率，又可以可视化验证准确率与学习率之间的函数曲线。根据函数曲线，缩小学习率范围，进行下一轮更精确的搜索。

3. 结论及其反思