APDP 项目集成 LLM4HGNAS 架构搜索方案（增强版）

目录

1. 现状分析

1.1 APDP 项目现有架构搜索模块

当前 APDP 项目（HAGCN-for-Solving-APDP）尚无架构搜索模块，核心结构为固定 Encoder + RL 训练：

```
nets/
├── attention_model.py    # 主模型（Encoder + Decoder）
├── graph_encoder.py      # 7 种异构注意力 + GCN
pdp/
├── problem_pdp.py        # APDP 问题定义
├── state_pdp.py          # 状态管理
train.py                  # REINFORCE 训练
eval.py                   # 评估
```

现有流程：

1. 固定 Encoder 架构
2. REINFORCE 训练

3. 验证 cost 并保存模型

1.2 LLM4HGNAS 的关键特性

```
LLM4HGNAS/
├── nas/search_space.py  # 搜索空间定义
├── hgnn/meta_manager.py # 模型管理与评估
└── nc_gpt.py            # GPT 驱动架构搜索
```

优势：

- 支持跨层连接（inter_layer）
- 更丰富的操作选项（zero_conv, gcn_conv, gat_conv, edge_conv, sage_pool）
- 迭代式搜索策略（早期探索，后期利用）

1.3 HAGCN 的强化学习训练

```
cost, log_likelihood = model(x)
bl_val, bl_loss = baseline.eval(x, cost)
reinforce_loss = ((cost - bl_val) * log_likelihood).mean()
loss = reinforce_loss + bl_loss
```

1.4 关键差异

特性	现有 HAGCN	LLM4HGNAS	目标方案
训练方式	强化学习	监督学习	强化学习 + NAS
架构搜索	无	LLM 驱动	LLM 驱动
搜索粒度	N/A	每层 + 跨层	每层 + 关系级别
问题类型	APDP 路径规划	节点分类/预测	APDP 路径规划

2. 方案概述

2.1 三种方案对比

方案	改动量	复杂度	优势	劣势
A	小	低	快速实现，风险低	搜索能力有限
B	中	中	功能完整，架构清晰	需要更多测试
C	大	高	性能潜力高，灵活性强	实现复杂

2.2 推荐方案

推荐 **方案 B**：在现有 HAGCN 框架上引入强化学习训练闭环，同时保持 LLM 架构搜索能力。

2.3 范围与目标

- 目标 1：用 LLM 生成可解析的 Encoder 架构，并完成动态构建。
- 目标 2：在 RL 训练中完成架构切换与性能反馈。
- 目标 3：形成可复现实验日志与最佳架构记录。

3. 方案 A：最小改动方案

3.1 核心思想

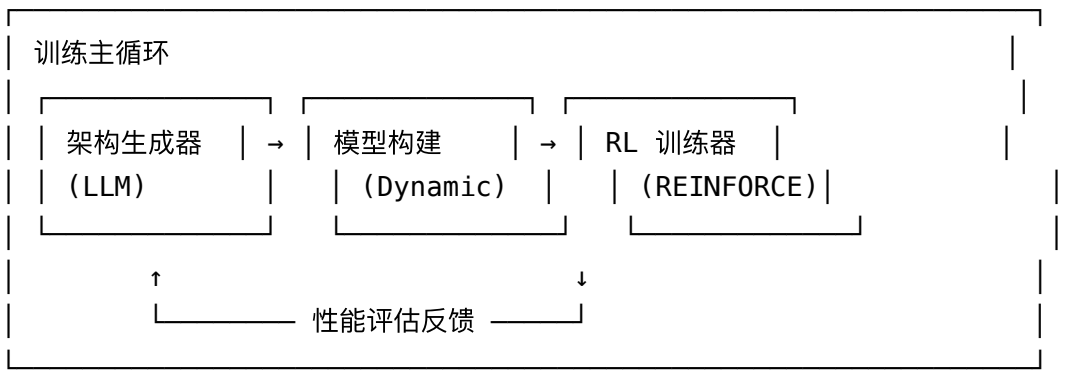
在 LLM4HGNAS 的评估流程中，将监督学习评估替换为 **短周期 REINFORCE 评估**，用 cost 作为架构得分。

4. 方案 B：完整集成方案

4.1 核心思想

创建新的训练流程：在强化学习训练过程中，每隔 K 个 epoch 切换一次架构，并记录性能。

4.2 系统架构



4.3 文件结构

```
LLM4HGNAS/
├── nas/search_space.py      # [修改] APDP 搜索空间
├── prompt.py                # [新增] apdp_prompt
├── nc_gpt.py                # [修改] 调用 HAGCN 评估

HAGCN-for-Solving-APDP/
├── nets/attention_model.py # [修改] 动态 Encoder
├── nets/graph_encoder.py   # [修改] 按 arch 构建层
├── train.py                 # [复用] RL 训练
```

5. 方案 C：双阶段训练方案

5.1 核心思想

分为两个阶段：

1. 架构探索阶段：快速评估多个候选架构
2. 训练阶段：使用最优架构进行完整 RL 训练

5.2 系统流程

```
阶段1：架构探索
for i in range(search_iterations):
  arch = LLM.generate()
  score = quick_rl_eval(arch, eval_epochs=5)

阶段2：完整训练
train(best_arch, epochs=100)
```

5.3 优缺点

优点：效率高、最终性能好；缺点：实现复杂，超参调优多。