

大模型与电网语义层的关系

Volt Mind 第一阶段明确坚持：“先不微调大模型，把资源投入电力语义层、工具系统、事件模型和专业算法。”

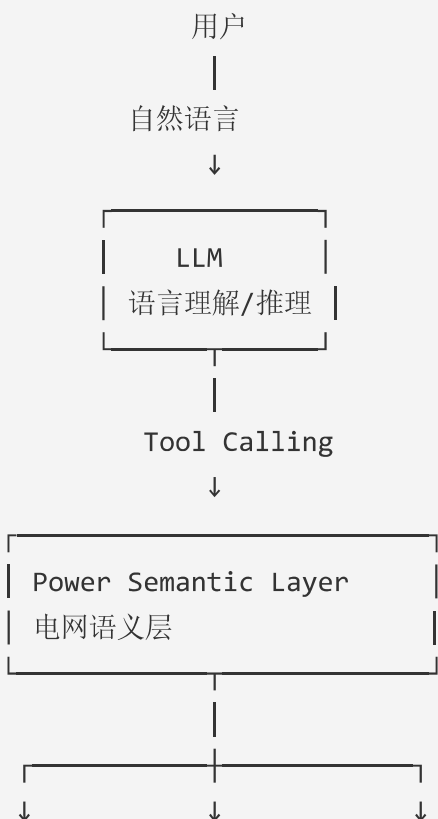
但有一个非常重要的细节需要区分：

不需要微调大模型 ≠ 不需要模型训练。

未来很可能需要训练**电力专业模型**，但不一定训练/微调那个负责自然语言交互的 LLM。

一、先说结论

我会把 Volt Mind 的 AI 架构拆成四层：



数据查询	专业工具	专业模型
OLAP	Event	Fault Model
MySQL	Engine	Risk Model
Graph	Rule	Anomaly Model

其中：

LLM负责

- 理解用户问题
- 判断用户意图
- 选择工具
- 组织工具参数
- 汇总工具结果
- 形成自然语言解释
- 多轮对话

电网语义层负责

- “馈线”是什么意思
- “设备”是什么意思
- `dev_oid` 对应什么设备
- 什么是“重复故障”
- 什么是“高风险馈线”
- 某个指标如何计算
- 哪些数据可以调用
- 哪些数据之间是什么关系

专业模型负责

- 遥测异常检测
- 故障分类
- 设备健康度
- 馈线风险预测
- 时序预测
- 相似故障计算

这是三个完全不同的问题。

二、为什么你说的“只需要自然语言 + 工具”在大多数场景下是成立的？

举个非常典型的例子。

用户问：

“帮我分析一下 XX 馈线昨天的故障。”

LLM并不需要知道：

XX馈线历史上发生过多少次故障

因为这些是**动态事实**，应该通过 Tool 获取。

LLM只需要知道：

用户要：

分析馈线故障

需要调用：

`get_feeder_fault_events()`

然后：

`get_soe_events()`

然后：

`get_telemetry_window()`

然后：

`get_device_profile()`

然后：

`get_similar_fault_cases()`

也就是说：

LLM ≠ 数据库
LLM ≠ 电网知识库
LLM ≠ 故障模型

LLM只是**大脑/编排器**。

三、实际上，电力语义层的价值就是把“专业知识”从 LLM 里面拿出来

这是我非常赞成你这个思路的原因。

传统路线可能是：

电力知识
↓
Prompt
↓
LLM

问题是：

Prompt 会越来越长，而且：

- 难维护
- 难验证
- 难追踪
- 难保证一致性
- 数据更新后容易过时

而 Volt Mind 应该做：

电力知识
↓
Power Semantic Layer
↓
Tools / API / Rules / Metrics

↓
LLM

例如：

用户问

“XX馈线近期是不是高风险？”

LLM 不自己计算。

它调用：

```
get_feeder_risk(  
    feeder="XX",  
    window="30d"  
)
```

Tool 返回：

```
{  
  "risk_level": "high",  
  "risk_factors": [  
    "30天故障3次",  
    "未消缺缺陷2项",  
    "设备异常次数上升",  
    "近90天重复故障"  
  ]  
}
```

然后 LLM 再把这些结果解释成人话。

这才是比较可靠的架构。

四、但是，有一种情况仍然可能需要微调 LLM

这里需要把“微调”拆成三个目的。

1. 为了让模型懂电力知识

通常不需要。

因为：

电力领域的实时事实、内部编码、局方规则，本来就应该放在语义层/知识库/工具里。

例如：

```
dev_oid = 8F923...
```

这种局方内部 ID，无论把模型训练多少次，也不是一个适合写进模型参数的知识。

2. 为了让模型更会“调用工具”

一般前期也不需要。

现代强 LLM 本身已经能够很好完成：

自然语言

↓

意图识别

↓

Tool selection

↓

参数生成

↓

结果解释

你们完全可以先通过：

- Tool schema
- System Prompt
- Few-shot
- Structured Output
- Workflow constraints
- Tool permission

解决。

例如：

工具：

```
get_feeder_faults(  
    feeder_id,  
    start_time,  
    end_time  
)
```

模型通常已经可以理解什么时候应该调用它。

所以第一代 Volt Mind 不应该一上来就：

“我们是不是需要训练一个电力大模型？”

我的答案是：

没必要。

五、第三种情况：为了“固定业务行为”

这个时候，微调才可能开始有价值。

例如经过大量真实运行以后，你发现：

同一个问题：

“为什么 XX 馈线跳闸？”

你希望模型总是按照固定顺序分析：

- ① SOE
- ② 保护动作
- ③ 遥信

- ④ 遥测
- ⑤ 拓扑
- ⑥ 历史案例
- ⑦ 缺陷
- ⑧ 给出原因
- ⑨ 给出处置建议

而不是有时候：

- ① 查历史案例
- ② 查缺陷
- ③ 忽略SOE

这时候可以考虑：

Fine-tuning / SFT

让模型更稳定地学习：

“电力故障分析应该怎样推理、怎样调用工具、怎样组织结果。”

注意，这时候微调学习的不是：

“XX变电站是什么。”

而是：

“遇到这类电力问题应该如何行动。”

这个区别非常重要。

六、我尤其建议你不要把“故障模型”和“LLM”混在一起

这可能是 Volt Mind 产品设计里面最重要的一点。

比如你们现在的 `fault_classification_data` 已经存在：

- `dev_oid`
- `remoteSignaling_id`
- `telemetry_ids`
- `level1`
- `level2`
- `model_version`

也就是说，它实际上已经有：

电气输入 → 故障分类结果

的结构。

这种东西非常适合做：

专业故障模型

比如：

Transformer / FTU / Feeder Fault Model

输入：

电流

电压

零序

功率

遥信

SOE

输出：

故障类别

故障概率

异常程度

疑似区段

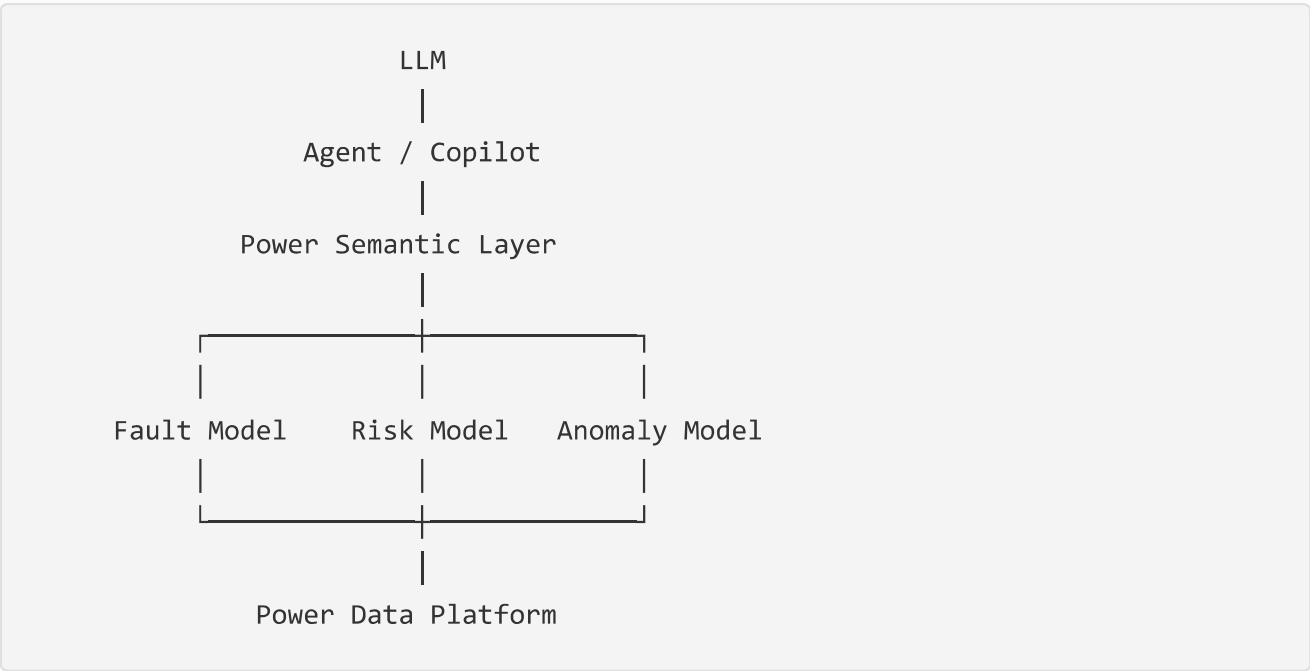
这个模型可以是：

- XGBoost
- LightGBM
- Transformer
- TCN
- 时序模型
- 图神经网络
- 其他专用模型

它和 LLM 是两个模型。

七、最终 Volt Mind 应该形成“1+N”模型体系

我比较推荐这个架构：



也就是说：

1个通用大模型

负责：

Understanding / Reasoning / Planning / Explanation

负责：

Detection / Prediction / Classification / Scoring

八、实际上，你们的竞争壁垒不应该是“微调了一个电力大模型”

这一点我非常想强调。

因为如果未来大家都可以买到：

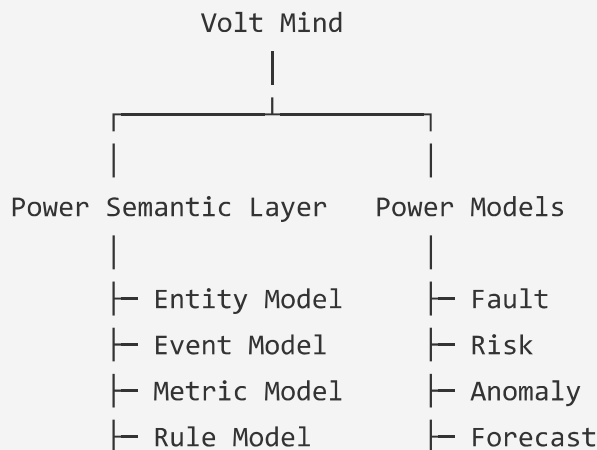
- GPT 类模型
- Claude 类模型
- 国产大模型
- 各种开源模型

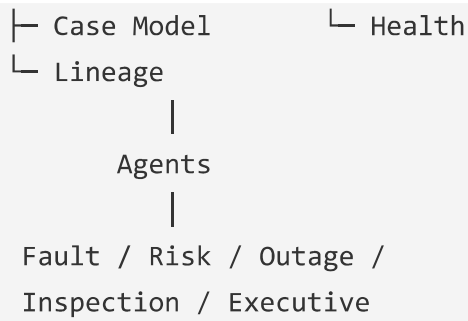
那么单纯：

“我们把 XX 大模型微调成了电力大模型”

这个壁垒并不牢固。

真正难的是：





这才是 Volt Mind 的真正产品。

九、甚至可以做到“大模型可替换”

我认为这是一个非常重要的产品战略。

比如今天用：

Model A

明天换：

Model B

后天换：

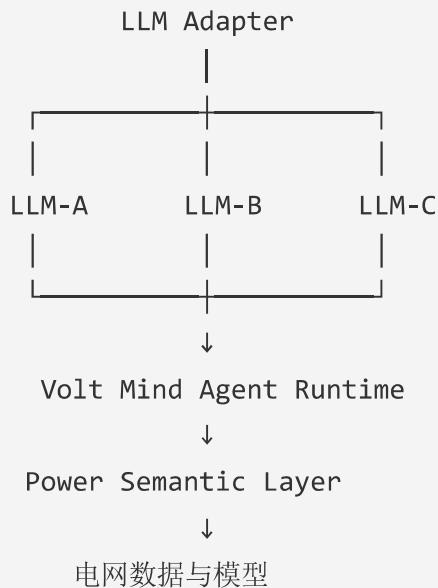
Model C

Volt Mind 的：

Power Semantic Layer
Agent Tools
Event Engine
Fault Engine
Risk Engine

都不应该变。

架构应该是：



这会让您们不被某一家基础模型厂商绑定。

十、那么 Volt Mind 到底什么时候值得做 LLM 微调？

我给你一个非常实用的判断标准：

问题	是否需要微调LLM
理解“馈线故障分析”是什么意思	×
识别用户想查询哪个馈线	×
调用 <code>get_feeder_faults()</code>	×
调用 SOE Tool	×
查询实时遥测	×
查询设备台账	×
查询历史案例	×
解释数据结果	×

问题	是否需要微调LLM
学习局方内部设备编码	×
学习实时电网事实	×
学习固定的 Agent Workflow	可能需要
提高复杂 Tool Calling 稳定性	可能需要
固定专业输出格式	可能需要
电力领域术语/语气高度专业化	可能需要
电力故障分类	不一定用LLM，优先专业模型
遥测异常检测	不应该靠LLM
负荷预测	应该用专业时序模型
设备寿命预测	应该用专业模型

十一、所以我对你们的建议其实非常明确

第一阶段

0% LLM Fine-tuning

把资源放在：

Entity Resolution
Semantic Layer
Event Engine
Tool System
RAG
Agent Workflow
Fault Model

第二阶段

开始收集：

用户问题

- Agent如何回答
- 调用了哪些Tool
- 工具结果
- 专家最终判断

形成自己的：

Power Agent Dataset

这个数据非常重要。

第三阶段

当数据积累到一定程度之后，再问：

“有没有必要微调？”

这时候答案可能是：

有。

但微调的目标已经不是：

让 LLM 学会电网知识。

而是：

让 LLM 更稳定、更低成本、更专业地完成 Power Agent 的工作流。

十二、我甚至会把 Volt Mind 的 AI 技术路线写成一句话

知识不进模型，进入语义层；事实不进模型，进入数据层；计算不交给大模型，交给专业模型；大模型负责理解、规划、调用、推理和解释。

我认为这句话非常适合成为 Volt Mind 的技术理念。

尤其你们现在这个地市局场景，已有实时遥测、遥信、SOE，以及故障、缺陷、停电、巡视、工作计划等多类业务数据。

所以你现在的理解方向是对的：先把“电力世界”做成机器可调用的 **Semantic Layer + Tools**，而不是急着把电力知识硬塞进 LLM 参数里。

更进一步说，我认为 **Power Semantic Layer** 本身很可能比“微调一个电力大模型”更值得成为 Volt Mind 的核心技术资产。