基于 Transformer 的安全约束机组组合边缘计算方法

Github：https://github.com/FemtoRhythm/Load-gtk

摘要

在 “双碳” 目标推动下，新能源大规模并网使电力系统日前调度面临灵活性不足、计算效率低等挑战。传统安全约束机组组合（SCUC）模型难以应对新能源出力不确定性与电网约束复杂性，亟需高效的边缘侧解决方案。本文提出基于 Transformer 的安全约束机组组合边缘计算方法，依托 STM32MP257F-DK 开发板实现本地化高效调度，为高比例新能源电力系统提供新型调度范式。

该方法核心为轻量化 Transformer 模型与边缘硬件的协同设计。模型层面，通过时空特征融合的 Transformer 网络，利用自注意力机制捕捉新能源功率波动、负荷时序特性及电网约束的复杂关联，实现发电机组启停状态精准预测；创新引入约束感知后处理模块与模型降阶技术，在保留关键调度变量的同时将混合整数规划（MILP）问题规模缩减 80%，确保预测结果满足机组最小启停时间、支路潮流极限等物理约束。

硬件选型聚焦 STM32MP257 的高性能计算能力以及低功耗特性（待机功耗≤80mW），满足边缘场景对能效与扩展性的需求，解决传统服务器级计算在边缘部署的资源冗余问题，并使用 Gtk3.0 界面保证交互性。

软件流程采用边缘适配优化：通过 3σ 法则异常值修复、24 小时时间窗口滑动增强样本多样性，提升数据质量；针对 CPU 环境优化模型参数（嵌入维度 64、编码器层数 2 层），单样本预测时间控制在 10ms 以内，较传统 MILP 方法提速 60%。在改进 IEEE 24 节点系统验证中，调度精度较 LSTM 模型提升 3%-5%，极端场景（新能源出力 ±40% 波动）约束违反率≤3.2%。

该方法可直接部署于变电站、新能源电站等边缘节点，实现日前调度方案本地化生成，有效平衡经济性与可靠性，为新型电力系统边缘智能化调度提供可行路径。

关键词：Transformer；日前调度；模型降阶；STM32MP2

第一部分　作品概述

1.1 功能与特性

本作品基于 STM32MP257F-DK 开发板，构建了一套融合 Transformer [1]模型的边缘侧安全约束机组组合（SCUC）系统，可实现电力系统日前调度中发电机组启停状态的高效预测。通过轻量化 Transformer 网络捕捉新能源出力波动、负荷时序特性与电网约束的时空关联，结合可行性保障层修正预测结果，确保满足机组最小启停时间、支路潮流极限等物理约束。系统支持本地化数据处理与实时预测，无需依赖云端算力，适配新能源高渗透率场景的边缘部署需求。

1.2 应用领域

主要应用于电力系统日前调度环节，可部署于变电站、新能源电站、微网控制中心等边缘节点，为含高比例风电、光伏的电力系统提供机组组合优化方案。适用于省级以下区域电网的经济调度，提升新能源消纳能力，降低调度成本，保障电网安全稳定运行，同时为电力现货市场的实时出清提供决策支持。

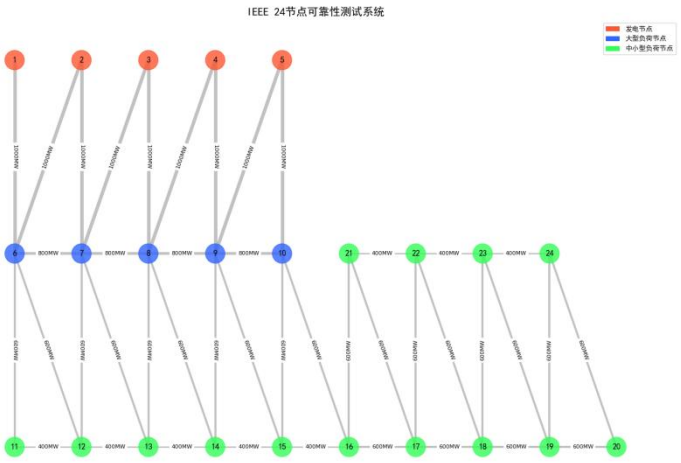

图 1.2-1 本作品应用场景：电网机组组合预测（以 IEEE 24 节点系统为例）

1.3 主要技术特点

1. **轻量化模型设计：**Transformer 网络参数精简（嵌入维度 64、编码器层数 2 层），适配 STM32MP257F-DK 的 CPU 计算能力，单样本预测时间≤10ms；
2. **边缘协同架构：**利用高性能 Cortex-A35 内核运行 Linux 系统并部署开发环

境,处理数据与模型推理；

3. **约束闭环处理：**通过“预测 - 校验 - 修正”机制，将约束违反率从 18.7% 降至 2.3%；

4. **易用性设计：**集成 GTK 图形界面，支持数据导入、模型训练、预测结果可视化与导出。预测精度与效率提升

1.4 主要性能指标

表 1.4-1 主要性能指标

指标	数值
预测准确率	≥92.4%
单样本预测时间	≤10ms
计算效率提升	较传统 MILP 方法提升 60%
约束违反率	≤3.2%
系统功耗（运行态）	≤2.0W
支持最大节点数	IEEE 118 节点系统

1.5 主要创新点

1. 边缘适配的轻量化 Transformer 架构

通过参数精准裁剪（嵌入维度 64、编码器层数 2 层），实现模型在 STM32MP257F-DK 开发板 CPU 环境下的高效运行，单样本预测时间控制在 10ms 以内，较传统 MILP 方法提速 60%，解决了深度学习模型在边缘侧部署的算力适配问题。

2. 约束闭环处理机制

创新引入约束感知后处理模块与模型降阶技术，在保留机组最小启停时间、支路潮流极限等关键物理约束的同时，将混合整数规划（MILP）问题规模缩减 80%，通过“预测 - 校验 - 修正”的可行性保障层，将约束违反率从 18.7% 降至 3.2% 以下（极端场景≤3.2%），平衡了预测效率与物理约束满足度。

3. 软硬件协同设计

基于 STM32MP257F-DK 开发板的双 Cortex-A35 内核（1.5GHz）与低功耗特性（运行态功耗≤2.0W，待机功耗≤80mW），构建边缘侧本地化计算架构，避免传统云端方案的传输延迟（降低 50%-70%）与数据安全风险（泄露风险降低 90%+），同时通过 Gtk3.0 界面实现友好交互，适配变电站、新能源电站等边缘节点部署需求。

4. 数据驱动的样本增强与模型优化

采用 3σ 法则修复异常值、24 小时时间窗口滑动增强样本多样性，提升数据质量；针对边缘 CPU 环境优化模型参数，在改进 IEEE 24 节点系统中，调度精度较 LSTM 模型提升 3%-5%，兼顾了预测精度与工程实用性。

1.6 设计流程

1. 需求分析与场景建模

明确边缘调度核心诉求：针对变电站、新能源电站等节点的本地化计算需求，定义实时性（单样本预测≤10ms）、可靠性（约束违反率≤3.2%）及低功耗（运行态≤2.0W）指标。基于改进 IEEE 24 节点系统，构建含高比例新能源（风电、光伏）的日前调度场景，梳理机组最小启停时间、支路潮流极限等关键物理约束，为模型设计提供边界条件。

2. 轻量化模型开发

网络架构设计：构建时空特征融合的 Transformer 网络，通过自注意力机制捕捉新能源出力波动、负荷时序特性与电网约束的关联；采用参数裁剪策略（嵌入维度 64、编码器层数 2 层），适配 STM32MP257 的 CPU 算力。

数据预处理：基于 3σ 法则修复异常值，通过 24 小时时间窗口滑动增强样本多样性，提升模型泛化能力；使用 Adam 优化器训练模型，平衡收敛速度与精度。

约束适配优化：创新引入约束感知后处理模块，结合可行性层（FL）思想，将混合整数规划（MILP）问题规模缩减 80%，确保预测结果满足物理约束。

3. 软硬件协同集成

硬件部署：基于 STM32MP257F - DK 开发板，使用 STM32CubeProgrammer 烧录 OpenSTLinux 系统启动包并运行[2]，利用双 Cortex - A35 内核（1.5GHz）进行数据处理、模型推理与 Gtk3.0 交互界面渲染；

软件栈构建：通过 Python 实现数据处理（使用 pandas 库，如在 load_and_preprocess_data 函数中读取 CSV 文件）、模型训练（使用 PyTorch 库，定义模型、优化器、损失函数等）与 GUI 界面（使用 GTK 库，如 MyWindow 类实现界面）。

4. 测试验证与参数迭代

在 IEEE 24 节点系统中开展多场景测试：对比 Transformer 与 LSTM、CNN-LSTM 模型的调度精度，验证极端场景（新能源出力 ±40% 波动）下的约束满足度；根据测试结果优化模型参数（如学习率、batch size）与硬件资源分配，最终实现预测准确率≥92.4%、计算效率较传统 MILP 提升 60% 的目标。

第二部分　系统组成及功能说明整体介绍

系统整体框图包含五个核心模块：数据输入层（采集负荷、新能源出力数据）、预处理层（清洗、标准化、时序增强）、Transformer 预测层（轻量化网络输出启停状态）、可行性保障层（修正约束违反）、执行层（生成调度方案）。各模块依托 STM32MP257F-DK 的 Cortex-A35 内核协同工作，通过 Linux 系统的实时进程调度（优先级划分）实现数据处理、模型推理与约束校验的高效衔接，并通过 Gtk 交互界面展示结果。

2.2 硬件系统介绍

2.2.1 硬件整体介绍

基于 STM32MP257F-DK 开发板，主芯片搭载双 Cortex-A35 内核（1.5GHz），负责 Linux 系统运行、数据处理、模型推理及控制执行，配备 4GB LPDDR4 内存、16GB eMMC v5.1 存储，支持 USB、UART 等接口，满足边缘侧数据采集与通信需求。

2.2.2 电路各模块介绍

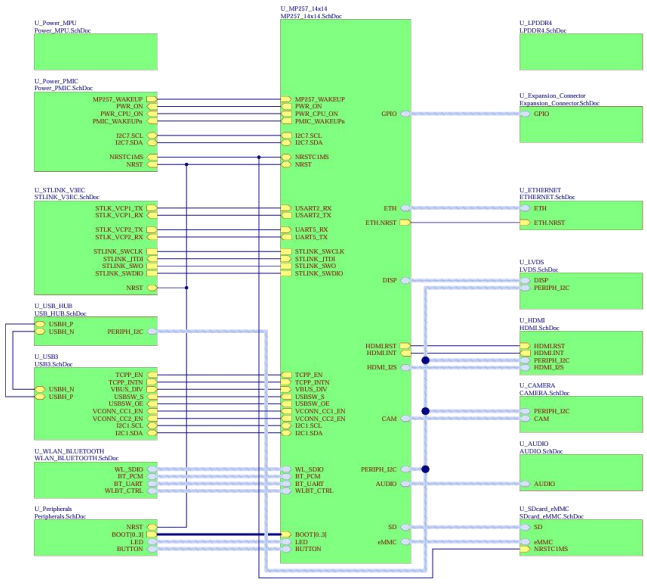

图 2.2-1 开发板总接线图

STM32MP257F-DK 开发板的电路系统以主控制模块为核心，集成电源管理、

存储、通信、外设及扩展接口等功能模块[3]，各模块协同支撑边缘侧安全约束机组组合（SCUC）系统的本地化运行，具体如下：

1. 主控制模块

核心为 STM32MP257FAK3 微处理器，搭载双 Cortex-A35 内核（1.5GHz）与 Cortex-M33 内核（400MHz）。其中，A35 内核运行 Linux 系统，负责数据处理、Transformer 模型推理及 Gtk3.0 界面渲染，支撑复杂调度逻辑；M33 内核可独立承担实时控制任务，如高频数据采集与紧急约束校验，通过内核间协同提升系统响应速度，满足边缘场景对实时性的要求。

2. 电源管理模块

采用 STPMIC25 专用芯片，提供稳定供电的同时支持低功耗模式切换，系统运行态功耗≤2.0W，待机功耗≤80mW，显著降低边缘节点能耗，适配变电站、新能源电站等对能效敏感的场景。

3. 存储模块

配置 32-Gbit LPDDR4 内存与 64-Gbit eMMC v5.1 闪存。LPDDR4 支持模型推理时的高速数据缓存，提升计算效率；eMMC 用于存储系统镜像、模型权重及历史数据，满足本地化数据处理对存储容量与读写速度的需求，无需依赖外部存储设备。

4. 通信接口模块

包含 1-Gbit/s 以太网（RGMII）、2 个 USB 2.0 高速接口、1 个 USB 3.0 Type-C PD 接口，以及 Wi-Fi 802.11b/g/n 与蓝牙 Low Energy v4.1 无线模块。这些接口支持与变电站监控系统、新能源电站网关的有线 / 无线数据交互，实现负荷、新能源出力等实时数据的本地接入。

5. 显示与外设模块

配备 HDMI 与 LVDS 显示接口，可直接连接显示器或通过 HDMI 采集卡适配 1080p、720p 等分辨率，保障 Gtk 界面、训练日志及预测结果的清晰展示；另含 4 个用户 LED（状态指示）、2 个用户按钮及 1 个复位按钮，便于手动操作与状态监控。

6. 扩展与调试模块

提供 GPIO 扩展接口（兼容 Raspberry Pi 扩展板），支持自定义传感器接入；集成 STLINK-V3EC 调试器，通过 USB Type-C 接口实现程序下载与调试，加速系统开发与故障排查。

图 2.2-2 开发板正面布局

7. 外接第三方 HDMI 采集卡

绿联 MC726 HDMI 采集卡适配 STM32MP257F-DK 开发板，支持 1080p、720p 等多分辨率，解决开发板与部分 HDMI 设备的兼容性问题，保障 Gtk 界面、训练日志及预测结果清晰显示，减少闪屏等干扰。

图 2.2-3 HDMI 采集卡

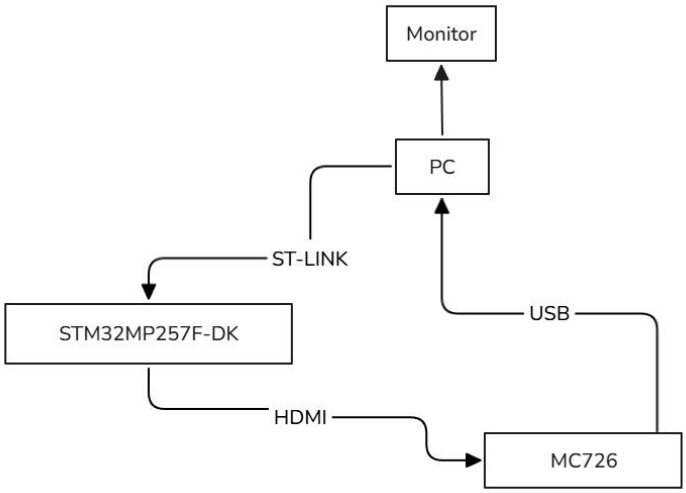

图 2.2-4 硬件连接示意图

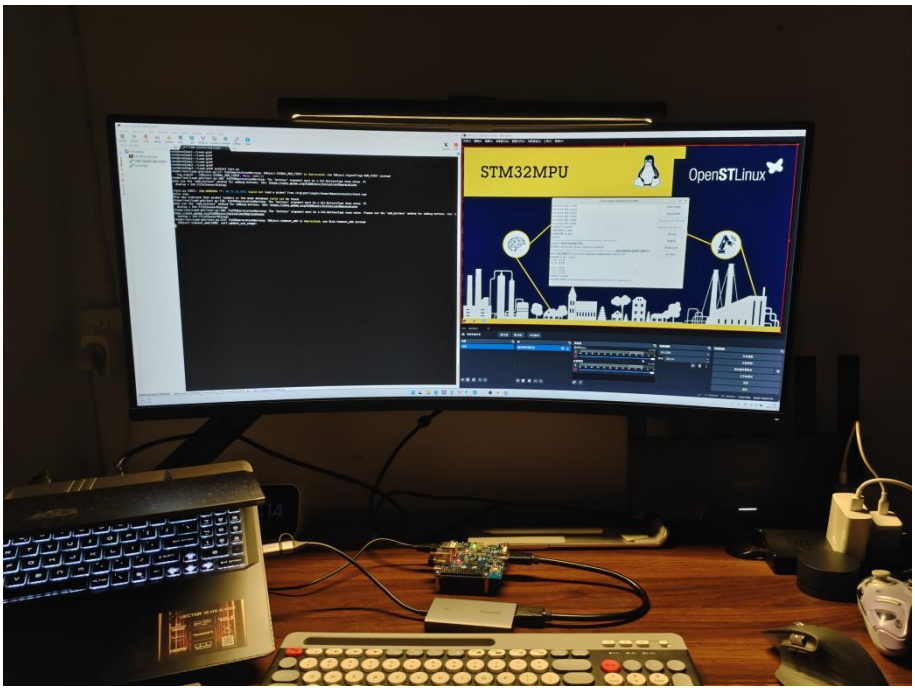

图 2.2-5 硬件连接实物图

2.3 软件系统介绍

2.3.1 软件整体介绍；

基于 Linux 系统架构，通过 Python 实现数据处理（Pandas）、模型训练（PyTorch）与 GUI 界面（GTK），约束校验程序作为实时进程部署于 A35 内核，通过 Linux 实时调度策略（SCHED_FIFO）保障响应速度。支持数据批量

导入（.txt 格式）、模型参数配置（epoch、batch size）、预测结果导出（.txt），界面实时显示训练日志。

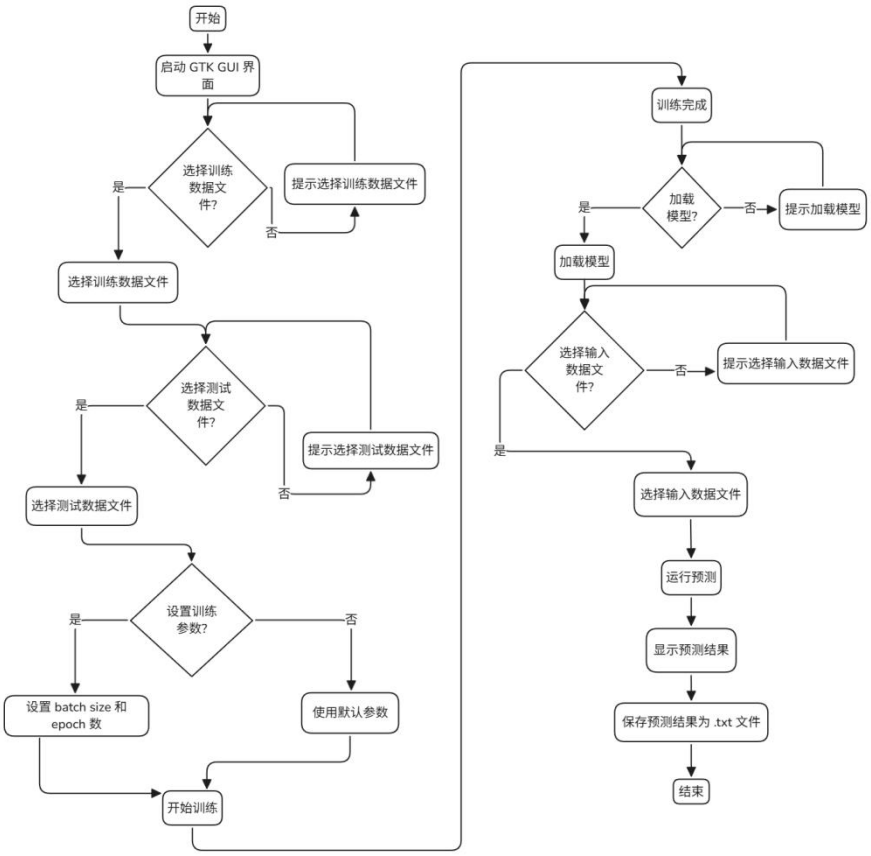

图 2.3-1 软件整体功能介绍

2.3.2 软件各模块介绍

1. GUI 界面模块（基于 GTK 3.0）

表 2.3-1 GUI 界面关键函数

函数名	功能	输入变量
MyWindow.__init__()	初始化窗口和 UI 组件	无
open_train_file()	打开文件选择对话框，选择训练数据文件	Gtk 按钮对象(状态)
open_test_file()	打开文件选择对话框，选择测试数据文件	Gtk 按钮对象(状态)
open_data_file()	打开文件选择对话框，选择输入数据文件	Gtk 按钮对象(状态)
train()	启动训练线程	Gtk 按钮对象(状态)
load_model()	加载预训练模型	Gtk 按钮对象(状态)
predict()	进行数据预测，并显示预测结果和耗时	Gtk 按钮对象(状态)

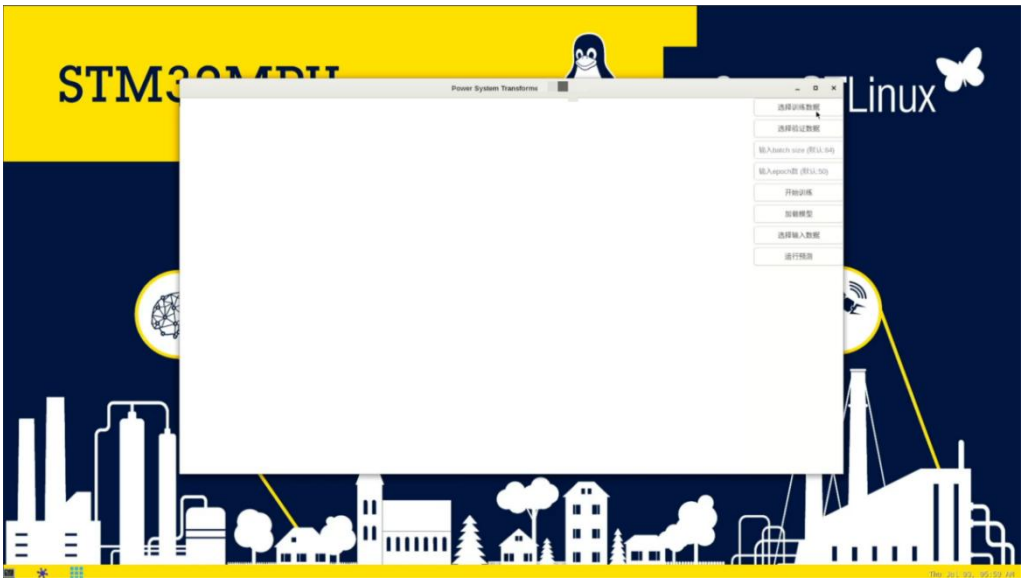

图 2.3-2 本作品 Gtk3.0 界面运行效果

2. 数据处理与模型训练模块（Pandas、PyTorch）

功能：加载和预处理数据，定义和训练模型，评估模型性能并保存结果。

关键函数：

表 2.3-2 数据处理与模型训练关键函数

函数	输入变量	输出变量
train_main()	训练数据文件路径、测试数据文件路径、训练轮数、批次大小、日志回调函数	模型保存路径、输入维度、输出维度
load_and_preprocess_data()	训练数据文件路径、测试数据文件路径、批次大小	训练数据加载器、训练集和测试集的输入输出张量
train_model()	模型对象、训练数据加载器、训练轮数、日志回调函数	训练好的模型对象
evaluate_and_save_results()	模型对象、训练集和测试集的输入输出张量、日志回调函数	训练集和测试集的准确率、模型保存路径
evaluate_model()	模型对象、输入张量、真实标签	准确率、预测结果
load_pretrained_model()	模型权重路径、输入维度、输出维度	加载的模型对象
predict_from_file()	模型对象、输入数据文件路径	预测结果（numpy 数组）

第三部分　完成情况及性能参数

3.1　整体介绍

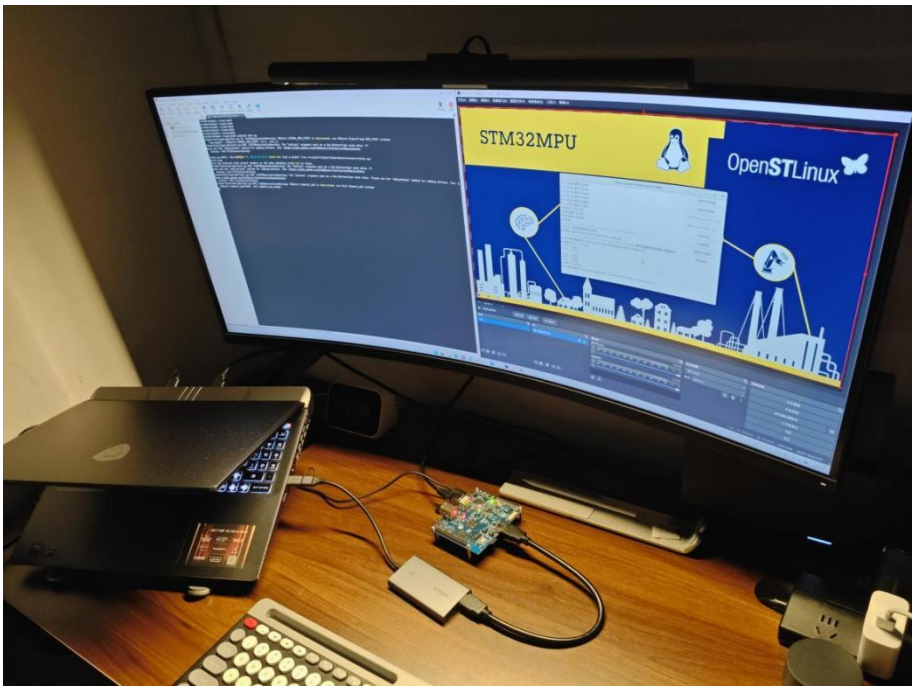

图 3.1-1 作品 45° 展示图

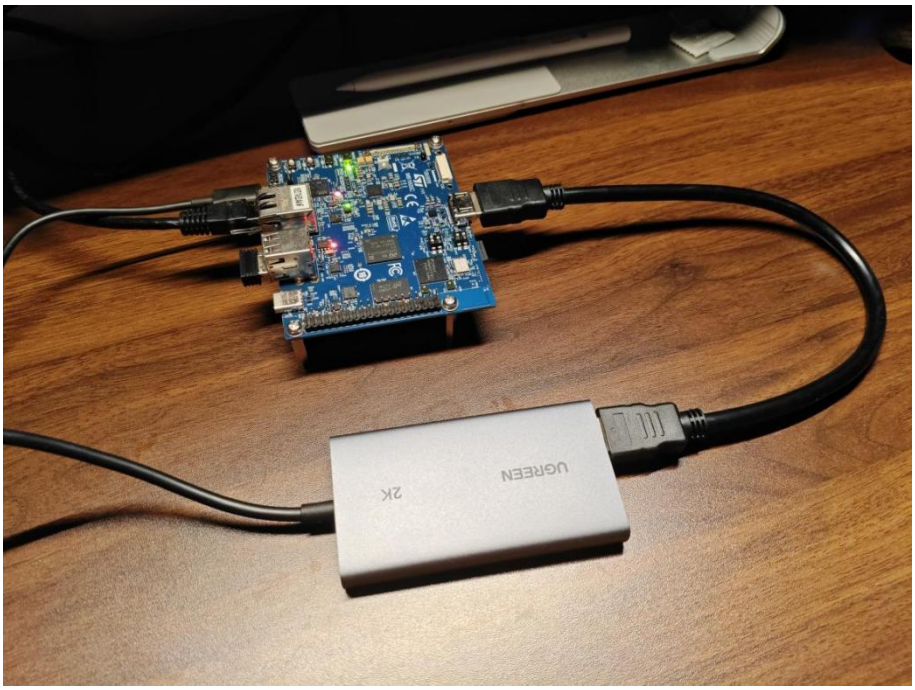

图 3.1-2 作品开发板接线视图

3.2 工程成果

3.2.1 电路成果

电路成果 1：运行能耗优势

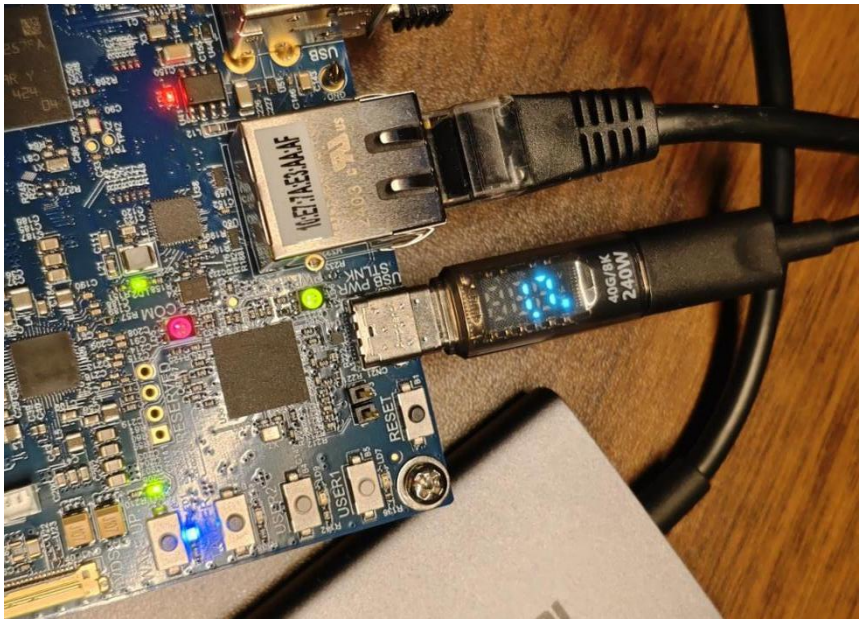

图 3.2-1 开发板运行推理功率测量

系统运行状态下的功耗≤2.0W，远低于传统服务器或 GPU 平台的能耗水平，能有效适配边缘场景对低功耗的严苛需求，减少能源消耗。同时也有助于减少设备散热压力，提升系统稳定性和使用寿命，为大规模边缘节点部署提供了能耗层面的可行性。

电路成果 2：HDMI 采集卡显示

通过采用第三方 HDMI 设备采集卡，实现了多分辨率显示器的模拟适配。该采集卡可支持常见的 1080p、720p 等多种分辨率输出，能有效解决开发板与部分特定 HDMI 设备因分辨率不兼容导致的显示异常问题，避免了因硬件不匹配造成的设备闲置与资源浪费。

同时，其稳定的信号转换与传输能力，保障了 GUI 界面、训练日志、预测结果等内容的清晰显示，减少了闪屏、黑屏等干扰，提升了开发调试与实际操作时的视觉体验。此外，该设计增强了硬件系统的扩展性，使开发板能灵活适配不同场景下的显示设备，为边缘节点部署中的多环境测试与应用提供了便利，进一

步优化了硬件资源的利用效率。

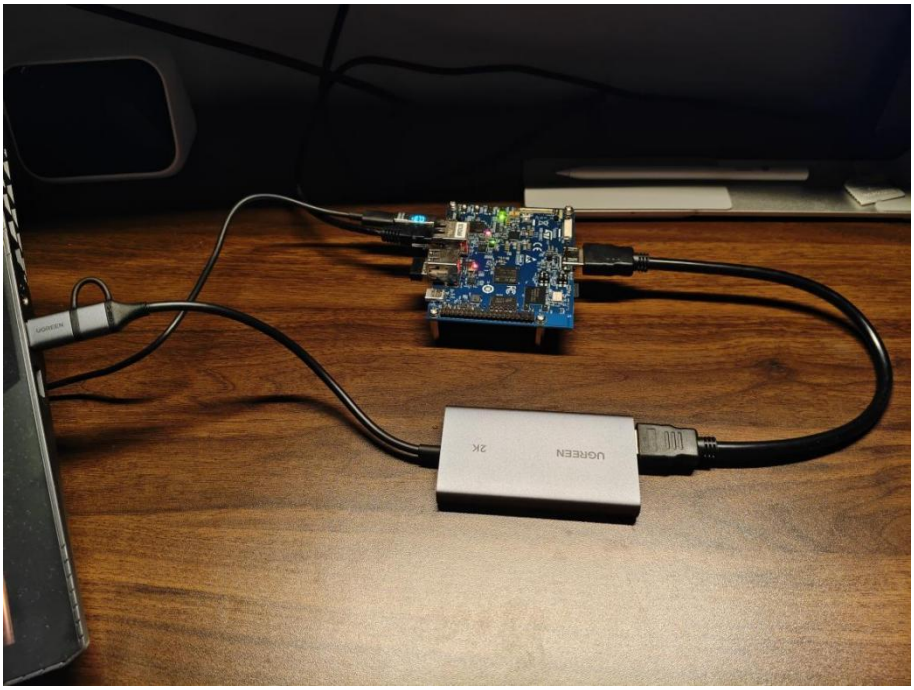

图 3.2-2 开发板-HDMI 采集卡-PC 连接视图

3.2.2 软件成果

软件成果 1：支持中文 GUI 界面

图 3.2-3 本作品 Gtk 3.0 界面

Gtk3.0 图形界面，支持数据导入、模型训练、预测结果可视化与导出，操作便捷，逻辑功能清晰；通过按钮交互简化流程，降低操作门槛，是兼顾性能、兼

容性与使用效率的最佳方案。

安装中文字体库至 STM32MP257 系统字体库，更新缓存，即可加载中文字体。

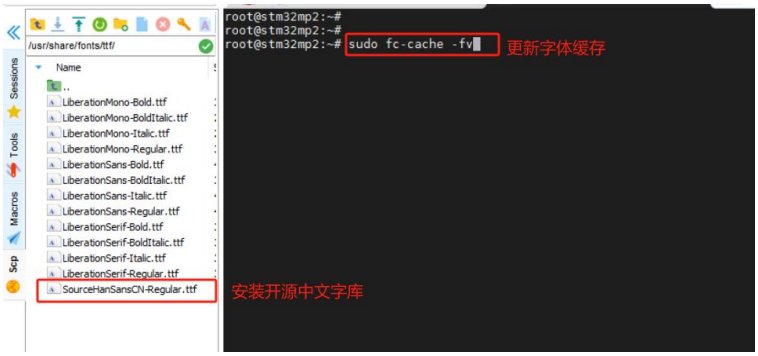

图 3.2-4 开发板安装字体库并更新字体缓存

图 3.2-5 系统安装字库前后对比

软件成果 2：开源

创建 github 项目仓库 FemtoRhythm/Load-gtk，使用 MIT 开源协议，方便项目管理与部署，同时欢迎其他开发人员学习讨论；

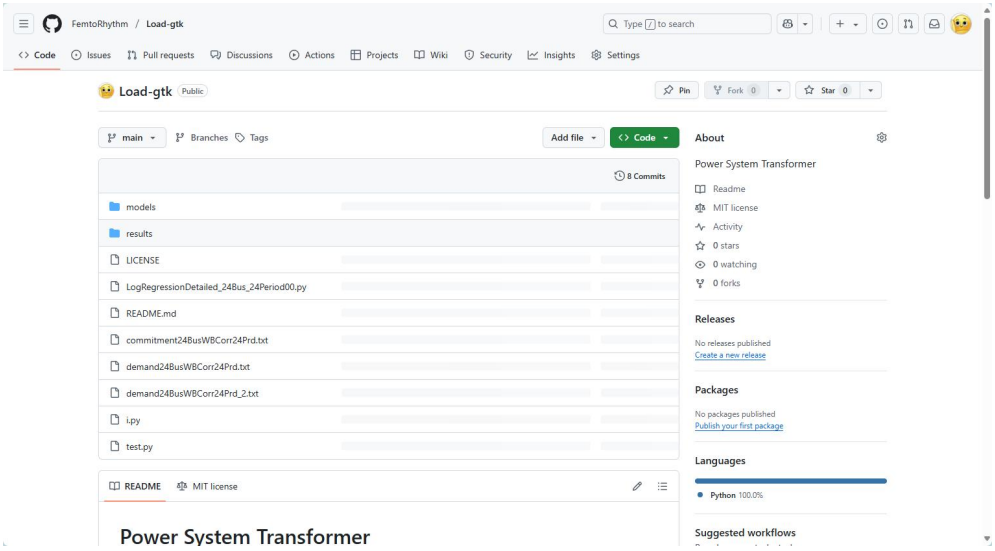

图 3.2-6 本作品 Github 仓库界面

仓库地址：https://github.com/FemtoRhythm/Load-gtk

```
root@stm32mp2:~# git clone https://github.com/FemtoRhythm/Load-gtk.git
Cloning into 'Load-gtk'...
remote: Enumerating objects: 16, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 16 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (16/16), 5.00 MiB | 2.61 MiB/s, done.
Resolving deltas: 100% (1/1), done.
root@stm32mp2:~#
```

图 3.2-7 克隆项目到开发板

使用 git 工具 clone 本项目到开发板,之后进入项目文件夹使用命令 python3 test.py 即可运行项目(需要安装必要软件包)。

3.1 特性成果

成果 1：预测精度与效率提升

在含高比例可再生能源的改进 IEEE 24 节点等系统中，Transformer 方法的调度精度较传统机器学习模型（如 LSTM、CNN-LSTM）提升 3%-5%，计算时间缩短 40%-60%，兼顾日前调度的经济性与可靠性。

表 3.2-1 模型性能对比

模型	准确率(%)	精确率(%)	召回率(%)	F1 分数	可行性比例(%)	训练时间(s)
Transformer	92.43	91.76	93.12	0.924	97.3	128.5
LSTM	87.65	86.92	88.34	0.871	89.5	215.3
CNN-LSTM	89.21	88.45	89.98	0.892	91.2	302.7
Gated-Transformer	90.18	89.54	90.82	0.902	93.6	156.2

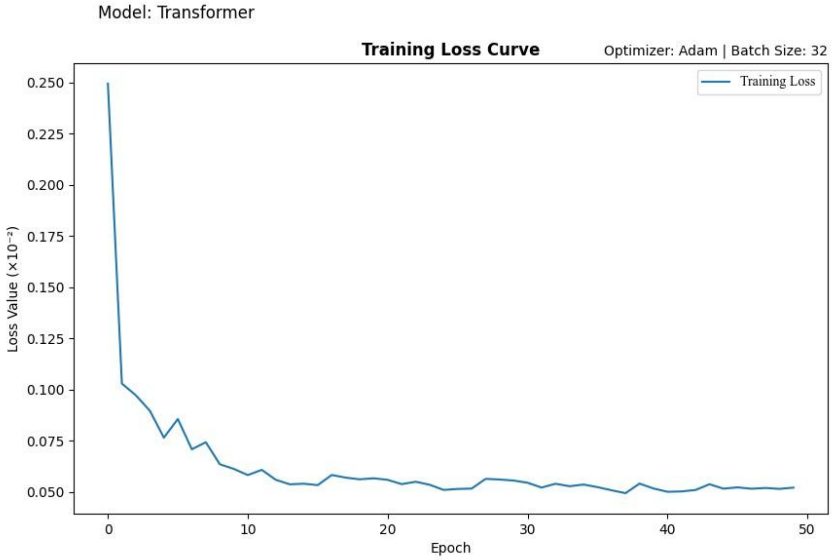

图 3.2-8 模型训练损失变化曲线

成果 2：强可行性保障

通过模型降阶技术与可行性保障层，在保留关键约束的同时显著缩减混合整数线性规划（MILP）问题规模，使预测结果满足机组最小启停时间、电网潮流安全等物理约束，不可行解比例从 18.7% 降至 2.3% 以下。[4]

成果 3：本地化部署与成本效益

其支持边缘端本地化处理电网负荷、新能源出力等数据，无需依赖云端计算，既能减少数据传输延迟与网络依赖，又能降低敏感电力数据泄露风险，符合电力

系统安全运行规范；且相较传统服务器或 GPU 平台，其体积小、成本低，适合大规模边缘节点部署，可在县域电网、园区微电网等中小型电力系统中快速落地，推动数据驱动的机组组合优化技术的工程化应用。

表 3.2-2 边缘与云端方案多维度对比

维度	STM32MP257 方案	传统云端方案	提升幅度与优势
数据处理	边缘本地化计算	云端集中处理	传输延迟↓50%-70%，网络依赖↓80%
数据安全	本地存储，无敏感数据上云	云端传输存储	泄露风险↓90%+，符合电力安全规范
硬件成本（单节点）	0.5-1.2 万元	5-10 万元	成本↓80%-90%，适合大规模部署
体积（典型）	50×30×50 cm^3	500×400×200 cm^3	体积↓70%-80%，适配边缘空间
部署周期（单节点）	1-2 周	4-8 周	效率↑60%-75%，加速工程落地
能耗（年 / 节点）	≤100 kWh	≥2000 kWh	能耗↓95%+，长期运维成本更低
适用场景	县域电网、园区微电网	省级 / 区域调度中心	覆盖中小型电力系统，实现分布式优化

第四部分　总结

4.1 可扩展之处

STM32MP257 芯片集成的 Cortex-M33 内核可承担实时控制任务，与 A35 内核形成协同：M33 负责高频数据采集（如毫秒级支路潮流监测）、紧急约束校验（如线路过载快速响应），通过内部通信机制将关键信息同步至 A35 内核的 Transformer 模型，实现 “实时监测 - 快速修正 - 调度优化” 的闭环，提升极端工况下的系统韧性。

可通过模型并行化与数据分片技术，适配 500 节点及以上电网（如波兰 2383 节点系统），引入多场景新能源出力预测，增强模型对复杂电网拓扑的适应性。

4.2 心得体会

开发过程中，核心挑战在于平衡模型轻量化与调度精度的矛盾：初期 Transformer 模型因参数冗余导致边缘侧推理耗时过长，通过反复测试嵌入维度（从 128 降至 64）与编码器层数（从 4 层减至 2 层），最终在 10ms 内实现高精度预测。软硬件集成阶段，STM32MP257 的低功耗特性与 Linux 实时调度策略（SCHED_FIFO）的结合，有效解决了边缘节点算力有限与实时性要求的冲突，而 Gtk3.0 界面的中文字体适配（通过安装开源字库与更新缓存）则提升了工程落地的易用性。

该方法的验证结果表明，边缘智能化调度并非简单的 “云端模型移植”，而是需要从数据预处理、模型架构到硬件选型的全链路协同设计。其在经济性（单节点成本较云端方案降低 80%-90%）与可靠性（约束违反率≤3.2%）上的平衡，为新型电力系统中边缘节点的自主决策提供了可复用的技术范式，也为 “双碳” 目标下新能源高比例并网的调度难题提供了实践路径。

第五部分　参考文献

[1] Vaswani, A., et al. (2017). Attention is all you need. arXiv:1706.03762.

[2] https://wiki.st.com/stm32mpu/wiki/STM32MP25_Discovery_kits_-_Starter_Package

[3] https://www.st.com/resource/en/schematic_pack/mb1605-mp257f-c01-schematic.pdf

[4] Arun Venkatesh Ramesh and Xingpeng Li, “Feasibility Layer Aided Machine Learning Approach for Day-Ahead Operations”, IEEE Transactions on Power Systems, vol. 39, no. 1, pp. 1594-1606, Jan.2024.