**主要功能**

1. **模型演示页面**
   * **硬件选择**：用户可以选择运行推理的硬件，如 CPU 或 GPU，确保模型推理时能够有效利用系统资源。
   * **模型选择**：用户可以从预定义的模型列表中选择想要运行的模型，或者加载自定义模型。
   * **输入选择**：支持多种数据类型作为模型输入（如图像、视频、文本等），并为每种输入类型提供相应的预处理功能（如图像缩放、归一化）。
   * **推理结果窗口**：提供一个可视化窗口实时显示推理结果，针对不同任务类型展示结果，如分类标签、目标检测的边界框、分割掩码等。
   * **模型运行数据**：显示与模型推理相关的关键性能数据，如推理时间、帧率、准确率等。
   * **CPU/GPU 使用监控**：集成系统资源监控工具，实时展示推理过程中 CPU 和 GPU 的使用率，便于用户了解硬件资源的消耗情况。
   * **错误信息窗口**：在推理过程中，若出现错误或模型加载失败，系统会弹出报错窗口，并给出详细的错误信息，帮助用户诊断问题。
2. **模型训练页面**
   * **模型框架选择**：用户可以选择不同的深度学习框架进行模型训练，如 TensorFlow 或 PyTorch。
   * **实时训练进度展示**：实时显示训练过程中损失值、准确率、学习率等信息，通过图表展示训练曲线，便于用户直观了解模型的训练进展。
   * **超参数调整**：提供图形化界面允许用户调整超参数（如学习率、批量大小等），并动态反映这些调整对训练过程的影响。
   * **CPU/GPU 使用监控**：集成硬件监控工具，实时展示训练过程中 CPU 和 GPU 的使用情况，帮助用户优化硬件资源利用率。
   * **暂停与恢复功能**：用户可以随时暂停或恢复训练，特别适合长时间训练的任务管理。
   * **模型验证与测试**：训练完成后，用户可以通过验证集和测试集评估模型的性能，并查看相关评估指标（如准确率、召回率、F1值等）。
   * **报错窗口**：在训练过程中，如遇到训练失败或框架错误，软件将显示详细的错误信息，并提示可能的解决方案。
3. **历史运行记录页面**
   * **数据表记录**：自动记录每次模型运行（无论是推理还是训练）的详细信息，包括运行时间、使用的模型、输入数据类型、推理/训练时长、模型准确率、损失值、错误信息等。
   * **历史对比功能**：支持对不同模型或同一模型在不同参数下的运行结果进行可视化对比，展示各个关键性能指标（如准确率、运行时长）的变化趋势，帮助用户做出合理的模型选择。
   * **日志导出**：用户可以将历史记录导出为 CSV 或 JSON 文件，方便后续的分析和分享。
   * **错误分类与解决建议**：软件会对不同类型的错误进行分类，如硬件故障、模型加载问题等，并提供相应的解决方案和参考资料。

**其他功能**

* **配置文件管理**：允许用户保存当前的模型、输入数据、超参数设置等，便于快速加载先前的实验配置并进行复现。
* **插件机制**：支持用户自定义模型和算法，便于开发者添加新的模型或功能模块，增强软件的可扩展性。
* **用户账户系统**：对于多人使用的环境，支持每个用户拥有自己的模型配置、历史记录等，确保数据的独立性与安全性。
* **自动化运行与报告生成**：用户可以选择多个模型批量运行，并生成报告，适合大规模实验和模型对比。
* **多语言支持**：提供多语言界面，适合国际化应用场景。

**技术架构**

* **前端界面**：基于 PyQt 的图形用户界面，提供简洁易用的操作体验。
* **后端模型处理**：支持 TensorFlow、PyTorch 等主流深度学习框架，集成 GPU 加速，确保高效的推理和训练过程。
* **数据存储**：历史运行记录与日志数据可以存储在本地数据库中（如 SQLite），确保数据的持久化和便捷管理。