基于无监督深度学习的工业缺陷检测系统的设计和实现

[1] 2 [2] 韩慧. 基于深度学习的工业缺陷检测方法研究[D]. 2019. [3] 赵君爱，基于图像处理的工件表面缺陷检测理论与方法研究[D]. 南京: 东南大学，2016. [4] 3 [5] 汤勃，孔建益，伍世虔. 机器视觉表面缺陷检测综述[J]. 中国图象图形学报，2017,22(12): 1640-1663. [10] 陆远蓉.基于UML的"4+1"视图软件体系结构描述研究[J].现代计算机(专业版),2011,(04):27-30.

摘要

产品质量是工业制造的基石，随着工业制造智能化进程的加速，产品质量检测的高效性与准确性需求日益提升，而缺陷检测正是工业产品质量检测中不可或缺的一环，它保障了各种工业制品的质量，如金属、芯片、纺织物等，在智能制造领域中扮演着重要角色。显而易见，传统的人工检测方法面临着效率低、成本高、稳定性不足等巨大问题，它注定会被更先进的方法取代。近年来，随着计算机视觉、工业成像、深度学习等领域的技术爆发，基于视觉的工业缺陷检测取得了突破性的进展，它为检测方法的更新迭代提供了新的可能性——构建神经网络对样本进行训练，提取特征，以实现自动化缺陷检测。然而，最初的有监督缺陷检测普遍依赖于大量标注的缺陷数据，这些标注数据不仅难以获得、成本高，且泛化能力相当有限，已不能满足现代化工业生产的需求，由此基于无监督深度学习的缺陷检测方法应运而生。无监督学习方法只需要提供正常样本，通过挖掘其特征就能实现缺陷识别，明显更适用于复杂的工业环境，逐渐成为研究的热点。然而，现有的无监督系统通常存在多尺度缺陷检测能力不足、功能模块割裂、参数配置复杂、实时性不足等问题，难以满足工业场景中快速部署的需求。为此，本研究基于开源算法SimpleNet和AnomalyGPT，结合工业检测实际需求，设计并实现了一套集样本管理、模型训练、缺陷检测模块于一体的无监督缺陷监测系统，旨在降低对标注数据的依赖，提升检测的灵活性，同时通过模块化设计与功能优化，提升系统的易用性与适应性。具体研究内容如下：

1. 针对样本类型多种多样、样本数量稀缺、样本质量参差不齐等问题，设计了由动态样本组管理、数据增强与样本编辑操作等部分构建的样本组管理模块。
2. 针对自监督缺陷检测依赖大量标注数据、模型训练参数设置复杂，以及系统模型管理困难等问题，设计了由基于SimpleNet的无监督缺陷检测功能、用户导向的参数映射功能与模型状态管理等部分构建的模型训练模块。其中，参数映射将复杂模型参数简化为"精度-速度-缺陷大小"等直观选项，降低操作门槛。
3. 针对检测结果仅通过阈值判定、检测结果单一、辅助信息有限等问题，设计了由基于AnomalyGPT的大模型辅助判定与交互功能、检测报告等部分构建的缺陷检测模块。系统通过热图可视化、大模型交互与检测报告，为工艺优化提供支持，具有一定的工业落地价值。

关键词：计算机视觉；无监督深度学习；工业缺陷检测；自动化检测；缺陷检测系统；

第一章 引言

1.1 研究背景及意义

在人类现代社会生活的各个方面，不论是衣食住行，亦或是晨昏四季，工业制品都无处不在。《中国制造2025》行动纲领指出，建设制造强国任务艰巨而紧迫，需要加速推进信息化与工业化的深度融合，推进生产过程的智能化[1]。众所周知，在工业制造智能化进程中，产品质量的把控始终是提升工业生产经济效益的关键环节，而这一环离不开产品的缺陷检测。通过缺陷检测能够有效把控产品质量、检测流水线机器的工作状态以及评估生产制造技术的优良，对提高产品质量和生产效率、降低生产成本有着至关重要的作用[2]。因此，基于视觉的工业缺陷检测不仅有非常重要的研究价值，同时也拥有广阔的应用前景[1]。在传统的工业生产过程，缺陷检测主要依靠人工视觉，不仅具有检测效率低、误检率和漏检率高、人工成本高、实时性差、主观误差高的缺点，还有接触损伤的风险。同时，在缺陷尺寸小于 0.5 mm 且无较大光学形变时，人眼检测不到缺陷信息，不适用于大规模工业生产的要求[3]。

后来，随着计算机图像处理技术的突破，机器视觉有效地解决了缺陷检测中人工的弊端。机器视觉检测技术是一种非接触式的自动检测技术，具有安全可靠、检测精度高、可在复杂的生产环境中长时间运行等优点，是实现工厂生产自动化和智能化的一种有效方法[4]。因而机器视觉逐渐取代人工视觉，成为工业缺陷检测的主力。目前，基于机器视觉的缺陷检测技术已广泛应用于工业产品的质量检测、分类检测和包装检测等，涉及钢板、玻璃、印刷、电子、纺织品、零件、木材、钢轨、瓷砖等多种关系国计民生的行业和产品[5]。

然而，基于规则的传统图像处理算法虽然在特定场景下效果稳定，但对于复杂纹理背景和多样化缺陷类型的适应能力较弱，需频繁调整参数。随着深度学习技术的发展，有监督学习方法通过其在理解和提取产品缺陷特征的优势，在检测精度和检测速度上取得了双重突破，但这类方法面临两大问题：一方面，工业场景中缺陷样本稀缺且多样性强，收集大量带标注的异常样本成本高昂；另一方面，工业生产过程的复杂性导致了缺陷模式和类型变幻莫测，有监督模型对未训练过的新型缺陷类型泛化能力有限，难以适应快速变化的生产工艺。

近年来，基于无监督学习的缺陷检测技术因其能自动学习潜在特征和模式，仅需正常样本即可完成训练的特性，逐渐成为研究热点。经研究发现，现有的无监督检测方法存在三个缺陷：第一，计算复杂度高，如部分基于生成对抗网络(GANs)的方法虽然检测精度较高，但运算开销大，难以满足工业实时检测需求；第二，参数配置专业性强，增加了工程部署门槛，不利于非专业用户使用；第三，检测结果信息量少，对于工业制造中的缺陷检测所能提供的辅助作用低下。此外，大多数现有方法仅提供异常评分而不能直接判断异常，需要人工设置阈值，这在动态变化的生产环境中缺乏灵活性。

本文旨在对基于无监督深度学习的缺陷检测算法模型如何向实际工业应用进行转化，以及相应缺陷检测系统的设计与开发展开研究。结合已有的开源算法模型，设计一套面向工业实际应用场景的无监督缺陷检测系统，致力于解决训练数据稀缺且质量参差不齐、模型检测速度与精度难以平衡、模型参数配置复杂、检测效益低下等问题，提高检测效率与准确性，降低检测成本与学习成本，帮助优化生产工艺，从而提高产品质量，提升生产制造的效益，最终实现促进工业智能制造的发展的目标。

1.2 国内外研究现状（引用核心论文或开源项目）

工业缺陷检测技术经历了从人工检测到计算机视觉自动检测的发展历程。随着深度学习技术的发展，缺陷检测方法也经历了从传统图像处理方法、有监督深度学习方法到无监督深度学习方法的演进。本节将对工业缺陷检测无监督深度学习方法的三个研究阶段进行展开。

1.2.1 传统图像处理方法阶段

在深度学习兴起前，主要依赖边缘检测、形态学运算、特征提取等传统方法。例如，通过阈值分割、边缘检测、形态学操作等技术实现缺陷区域提取。这些方法对简单场景和规则缺陷有效，但在复杂场景下性能有限，且需频繁调整参数。

1.2.2 有监督深度学习方法阶段

随着深度学习技术的发展，卷积神经网络(Convolutional Neural Networks, CNN)成为缺陷检测的主要工具，有监督学习方法开始占据工业缺陷检测领域的主导地位。如Faster R-CNN、YOLO系列算法在缺陷定位与分类中取得显著进展，能够在大量标注数据上实现高检测准确率。然而，这些方法需要大量标注数据，而工业缺陷数据难以获取且成本高昂。

1.2.3 无监督深度学习方法阶段

为解决标注数据不足的问题，研究人员开始采用无监督方法，如基于重构误差或特征学习的方法，学习正常样本的分布来检测异常样本。这些方法仅需正常样本即可完成训练，大幅降低数据获取成本。

1.4 研究内容与主要工作

本研究围绕无监督学习的缺陷检测系统的设计与实现展开，基于已发布的开源无监督检测算法SimpleNet和缺陷检测大语言模型AnomalyGPT，重点解决现有系统在用户可用性及易用性、工程落地效益及可行性等方面的不足。核心研究内容与主要工作分为以下四部分：

无监督检测算法优化与工业适配
针对工业场景中缺陷样本稀缺的难题，采用以SimpleNet为核心的无监督学习框架，优化算法以适应复杂工业环境。研究重点包括优化多层级特征提取策略，融合卷积神经网络中不同层级的全局与局部特征，增强对微小缺陷的敏感度；设计动态参数调整机制，根据缺陷尺寸与图像分辨率自适应调节特征采样率及比对阈值，实现精度与速度的平衡；开发伪缺陷生成技术，通过模拟亮度异常、随机噪点注入等方式扩充训练数据，提升模型对未知缺陷类型的泛化能力。

多尺度缺陷检测与轻量化工程实现
为满足工业场景中多样化缺陷的检测需求，构建多尺度检测体系。通过跨层特征金字塔网络整合宏观表面异常与微观局部缺陷的检测能力；采用模型剪枝、量化压缩及TensorRT加速技术，降低模型计算复杂度，在保证检测精度的同时将推理速度提升至单图0.2-0.5秒；设计客户端-服务器异步架构，分离用户交互与高负载计算任务，利用多线程优化提升系统吞吐量，适配高频产线实时检测需求。

用户友好交互系统设计与开发
针对非专业用户的操作痛点，设计直观易用的交互系统。开发参数映射模块，将专业算法参数（如特征维度、采样率）转换为"精度-速度-缺陷大小"等直观选项，降低配置复杂度；集成可视化功能模块，提供训练进度实时监控、缺陷热图叠加显示及分级报告自动生成功能，增强结果可解释性；基于PySide6框架开发跨平台桌面客户端，支持Windows与Linux系统，实现项目管理、样本导入、模型训练与检测的一站式操作流程。

系统验证与工业场景适配性优化
通过多维度实验与实际部署验证系统性能。在公开数据集（MVTec AD）与自建工业数据集（涵盖PCB板、金属表面等场景）中测试检测精度与鲁棒性，对比传统阈值分割、有监督模型及商业系统的性能差异；针对真实工业环境中的光照波动、设备震动等干扰，集成图像去噪模块与稳定性增强算法，优化系统在复杂条件下的适应性；开展产线试点部署，收集实际检测数据并迭代优化模型，确保系统在动态生产环境中的可靠运行。

创新性成果
提出低数据依赖的无监督检测框架，仅需正常样本即可完成模型训练；构建多尺度自适应检测机制，实现从宏观异常到微观缺陷的全覆盖；设计用户导向的轻量化系统架构，通过参数映射与工程优化降低落地门槛。本研

究通过算法改进、工程实践与用户体验设计的深度融合，为工业质检智能化提供了一套高效、易用且可扩展的解决方案。

1.5 论文组织结构

本文围绕无监督学校的缺陷检测系统的设计与实现展开，全文共分为六章，具体组织结构如下：

第一章 引言
阐述工业质检智能化转型的背景与挑战，分析传统检测方法及有监督学习技术的局限性，明确无监督学习在缺陷检测中的研究价值。梳理缺陷检测领域的技术发展脉络，对比分析国内外在传统算法、有监督学习及无监督学习方向的研究，总结现有方法的优势与不足，为本研究的突破方向提供理论依据。最后提出本研究的目标与创新点，并概述论文整体结构。

第二章 基本概念和相关工作
介绍无监督异常检测的基本概念，包括特征空间、记忆库构建、特征匹配等。分析SimpleNet算法原理，包括特征提取、记忆库构建、异常检测等核心步骤。

第三章 系统需求分析与总体设计
结合工业质检场景的实际需求，从功能性与非功能性角度定义系统设计目标。使用客户端-服务器分离架构，明确各模块的交互逻辑与数据流，为后续实现奠定框架基础。

第四章 系统实现与工程化部署
描述系统参数映射模块、可视化模块、模型推理优化及数据管理模块的设计与实现，涵盖客户端交互界面开发、服务器端接口设计。最后集成为基于PySide6的跨平台应用。

第五章 结果展示与分析
展示使用的开源算法SimpleNet以及AnomalyGPT的检测结果，并简要说明为何使用该算法。然后展示本系统的应用效果，并说明本系统在工业质检应用中的优势。

第六章 总结与展望
总结本研究的成果，分析其局限性，并提出未来研究方向，包括多算法适配、有监督-无监督混合学习、多模态数据融合及边缘计算部署等拓展路径。

1.6 本章小结

2.2.2 DeepVision

2.3 本章小结

第三章 关键技术分析

3.1 SimpleNet

3.1.1 算法核心思想

3.1.2 算适用场景与局限性

3.1.3 选择该算法的原因

3.2 AnomalyGPT

3.1.1 大模型原理

3.1.2 适用场景与局限性

3.1.3 选择该模型的原因

3.3 技术栈

3.3.1 开发框架

pyside6、Qt

3.3.2 数据库

mysql

3.3.3 部署工具

第四章 系统设计

4.1 总体描述

4.1.1 系统目标

4.1.1 总体描述

系统目标

本系统属于工业质检的辅助工具，旨在帮助用户快速定位工业产品表面的缺陷区域，提高检测效率。作为一款跨平台的轻量级桌面应用程序，前端使用 Qt 框架和 PySide6 库进行开发，后端则采用 FastAPI 框架和 MySQL 数据库。系统主要分为样本管理模块、模型训练模块、缺陷检测和报告模块。该应用具有样本编辑管理、模型可视化训练、实时缺陷检测、AI智能判别和自动报告生成等功能，能辅助工业质检人员方便、快捷、直观地识别各类工业产品的表面缺陷，同时根据用户需求平衡检测精度和速度，适应不同尺寸的缺陷检测场景。

该系统的算法核心主要基于SimpleNet和AnomalyGPT，能够实现仅基于正常样本的无监督学习、缺陷检测和智能化结果解释。系统能够提供用户友好的界面，如参数配置界面，同时支持专业用户与业余用户操作。特别地，该系统应该尽量满足人机交互启发式原则，如系统状态可见度、一致性和标准化等。另外，该系统应能够支持多线程处理，以保证高效的数据处理能力。

4.1.2 用户特征

本系统的目标用户是工业质检人员和无监督学习缺陷检测的研究人员，用户的需求是通过使用本系统来高效检测工业产品表面缺陷，提高质检效率并降低人工成本。考虑到工业质检人员的工作背景，部分质检员虽然具备丰富的视觉检测经验，但对深度学习模型的理解和参数配置缺乏专业知识，并且传统检测方式已形成习惯，对计算机辅助检测系统的使用存在一定门槛。研究人员则需要灵活配置系统参数并获取详细的检测结果数据。因此，本系统应尽量简化操作流程，提供直观的参数设置界面，同时为不同需求的用户提供基础到高级的功能选项，既满足质检员快速上手的需求，又能支持研究人员进行深入的算法优化与分析。

4.2 需求分析

4.2.1 功能性需求

1. 项目管理功能。用户可以创建、打开和管理缺陷检测项目，保存项目元数据和配置信息。此外，系统还提供历史项目快速访问功能，支持项目导航和拖放打开项目等操作，方便用户快速恢复之前的工作状态。

2. 样本管理功能。用户可以创建和管理多个样本组，批量导入和组织样本图像。系统支持图像预处理（裁剪、旋转、缩放）和数据增强（几何变换、亮度调整、颜色变换），还可通过伪缺陷生成功能模拟各类缺陷特征以丰富训练数据集。用户完成样本准备后，可将样本上传至服务器进行后续处理。

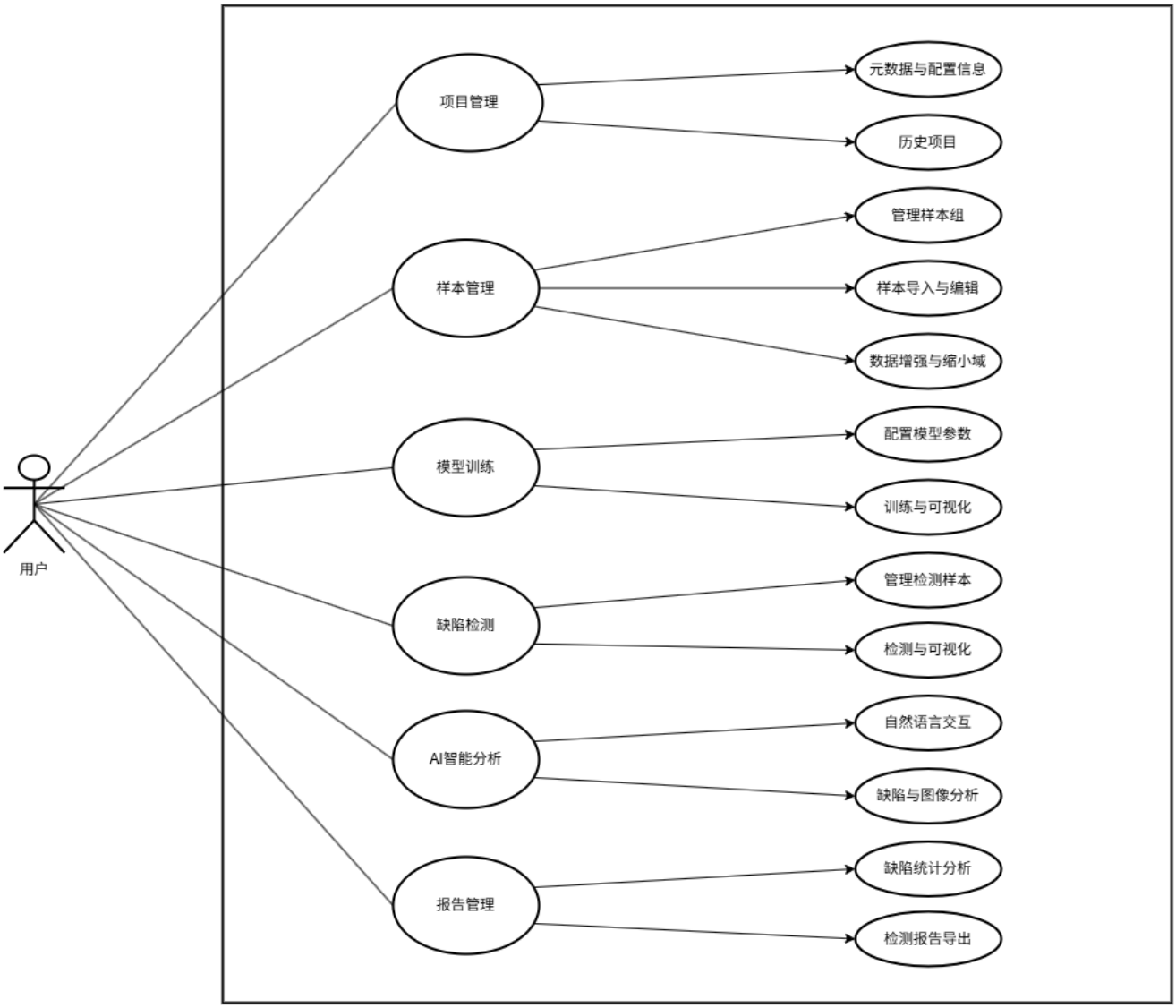
3. 模型训练功能。用户可以通过直观的参数映射系统配置模型，无需深入了解算法细节。系统将专业参数（如嵌入维度、层数、补丁大小等）转换为"精度-速度-缺陷大小"等选项，同时提供训练过程可视化，实时显示训练进度和性能指标。

4. 缺陷检测功能。用户可以导入和管理待检测样本，使用已训练的模型执行缺陷检测。系统集成服务器检测能力，提供结果可视化功能，包括原图/热图切换、缺陷区域标记等。用户可调整检测阈值优化结果，并进行批量检测和结果存储。

5. AI辅助分析功能。用户可通过自然语言交互界面对检测结果进行智能分析。系统集成大模型分析能力，支持多图像分析，包括说明缺陷是否存在（无需手动设置阈值）和缺陷位置，以及提供图像相关信息，使检测结果更具可解释性和实用价值。

6. 检测结果报告功能。用户可查看缺陷数据统计分析，包括位置分布、聚类分析等多维度信息。系统自动生成检测报告，集成各类可视化图表，并支持导出为PDF格式，便于分享和存档。报告内容既包含技术数据，也提供针对工艺改进的实用建议。

系统用例图如图4-1所示。



4.2.2 用例描述

1. 缺陷检测项目管理用例

名称	**内容描述**
ID	01
用例名称	项目管理
参与者	用户
描述	允许用户创建新项目、打开已有项目、管理项目元数据和配置信息
触发条件	用户启动系统或选择项目管理功能
前置条件	系统正常运行
后置条件	创建项目或项目信息被用户修改时，系统自动保存
优先级	低

名称	内容描述
正常流程	1. 用户启动系统，显示项目管理界面 2. 用户选择"新建项目"或"打开项目"或"最近项目" 3. 新建项目时，用户输入项目名称和存储位置 4. 打开项目或最近项目时，用户选择本地存储的项目文件 5. 系统加载项目，初始化配置信息
扩展流程	3a. 如果项目名称已存在或非法，提示用户选择其他名称 4a. 如果项目文件损坏，显示错误信息 4b. 用户可通过拖放方式打开项目文件 4c. 如果最近项目不存在，自动从历史列表中删除
特殊需求	无

2. 训练样本管理用例

名称	内容描述
ID	02
用例名称	样本管理
参与者	用户
描述	允许用户创建样本组、导入样本图像、进行预处理和数据增强，以及上传至服务器
触发条件	用户进入样本管理流程
前置条件	已创建或打开项目
后置条件	样本上传至服务器
优先级	高
正常流程	1. 用户创建新样本组或导入已有样本组 2. 用户导入图像文件或文件夹 3. 系统显示导入的样本列表 4. 用户可选择图像进行裁剪、缩小域等预处理 5. 用户可对样本进行数据增强或添加伪缺陷 6. 用户将样本上传至服务器
扩展流程	2a. 如果导入非图像文件，系统过滤不支持的文件 4a. 用户可批量应用部分预处理操作 6a. 大批量图像处理时显示进度条 6b. 上传失败时显示错误信息
特殊需求	无

3. 无监督学习模型训练用例

名称	内容描述
ID	03

名称	内容描述
用例名称	模型训练
参与者	用户
描述	允许用户配置和训练无监督缺陷检测模型，提供训练可视化和参数管理
触发条件	用户进入模型训练流程
前置条件	已上传训练样本至服务器
后置条件	模型训练完成
优先级	高
正常流程	1. 用户创建模型组或导入已有模型组 2. 用户选择要训练的样本组 3. 用户通过直观界面配置模型参数 4. 用户启动训练过程 5. 系统显示训练进度和实时性能指标 6. 训练完成后，系统修改模型状态并通知用户
扩展流程	3a. 用户可选择自定义参数配置 4a. 训练过程中用户可手动终止训练
特殊需求	无

4. 无监督缺陷检测用例

名称	内容描述
ID	04
用例名称	缺陷检测
参与者	用户
描述	允许用户导入检测样本，使用训练好的模型进行缺陷检测，并查看实时检测结果
触发条件	用户进入缺陷检测流程
前置条件	已选择或导入检测样本组；模型处于空闲状态，如训练完成或检测完成
后置条件	无
优先级	高
正常流程	1. 用户创建检测样本组并导入待检测图像 2. 用户选择处于空闲状态的模型 3. 用户启动检测过程 4. 系统执行检测并展示实时结果 5. 检测完成后，模型状态还原为空闲

名称	内容描述
扩展流程	1a. 支持批量导入检测样本 3a. 用户可调整阈值优化结果显示 5a. 用户可切换原图和热力图视图
特殊需求	无

5. AI辅助检测分析用例

名称	内容描述
ID	05
用例名称	AI智能分析
参与者	用户
描述	允许用户通过自然语言交互获取缺陷分析结果，
触发条件	用户在缺陷检测前后选择AI辅助分析
前置条件	已导入检测样本
后置条件	无
优先级	中
正常流程	1. 用户选择需要分析的检测样本 2. 系统加载大模型分析功能 3. 用户通过自然语言界面提问 4. 大模型分析图像并提供缺陷判断、位置描述和图像相关信息等 5. 用户可进行多轮对话深入了解分析结果
扩展流程	3a. 用户可同时选择多张图像进行对比分析 5a. 用户可保存分析会话记录
特殊需求	无

6. 缺陷检测结果报告用例

名称	内容描述
ID	06
用例名称	检测结果报告
参与者	用户
描述	允许用户查看缺陷统计分析，生成和导出报告
触发条件	用户选择结果分析与报告
前置条件	已完成缺陷检测并有结果可供分析
后置条件	无

名称	内容描述
优先级	中
正常流程	1. 用户选择检测样本组 2. 用户配置报告参数，如聚类参数和块大小 3. 系统执行缺陷结果统计与聚类分析，生成缺陷分布热图和统计图表 4. 用户查看分析结果 5. 用户选择生成报告 6. 系统生成PDF格式报告
扩展流程	无
特殊需求	无

4.2.3 非功能性需求

易用性：系统应尽量满足人机交互启发式原则，如针对部分操作给予系统反馈、界面排版布局的风格一致和标准化、针对用户可能出现的误操作采取限制等。系统界面应该简洁美观，同时操作方便，易于非专业用户上手。

可靠性：系统应能稳定运行，并提供错误处理机制，当系统出现错误时，如用户操作不当、服务器故障、网络通信问题等，系统应自动恢复并继续运行，不能崩溃。

性能：单张图像的检测时间不应超过 1 秒，用户界面操作响应时间不应超过 0.5秒，复杂操作应显示处理进度，避免用户长时间等待。

可维护性：系统应能方便地进行维护和更新，如添加新功能、修复已知问题、优化性能等。

可扩展性：系统应采用模块化设计，新功能模块的添加不应影响其他模块的正常运行，便于扩展。同时，系统应对检测算法解耦，高内聚低耦合，便于日后在不修改核心架构的基础上集成更多的无监督学习算法。

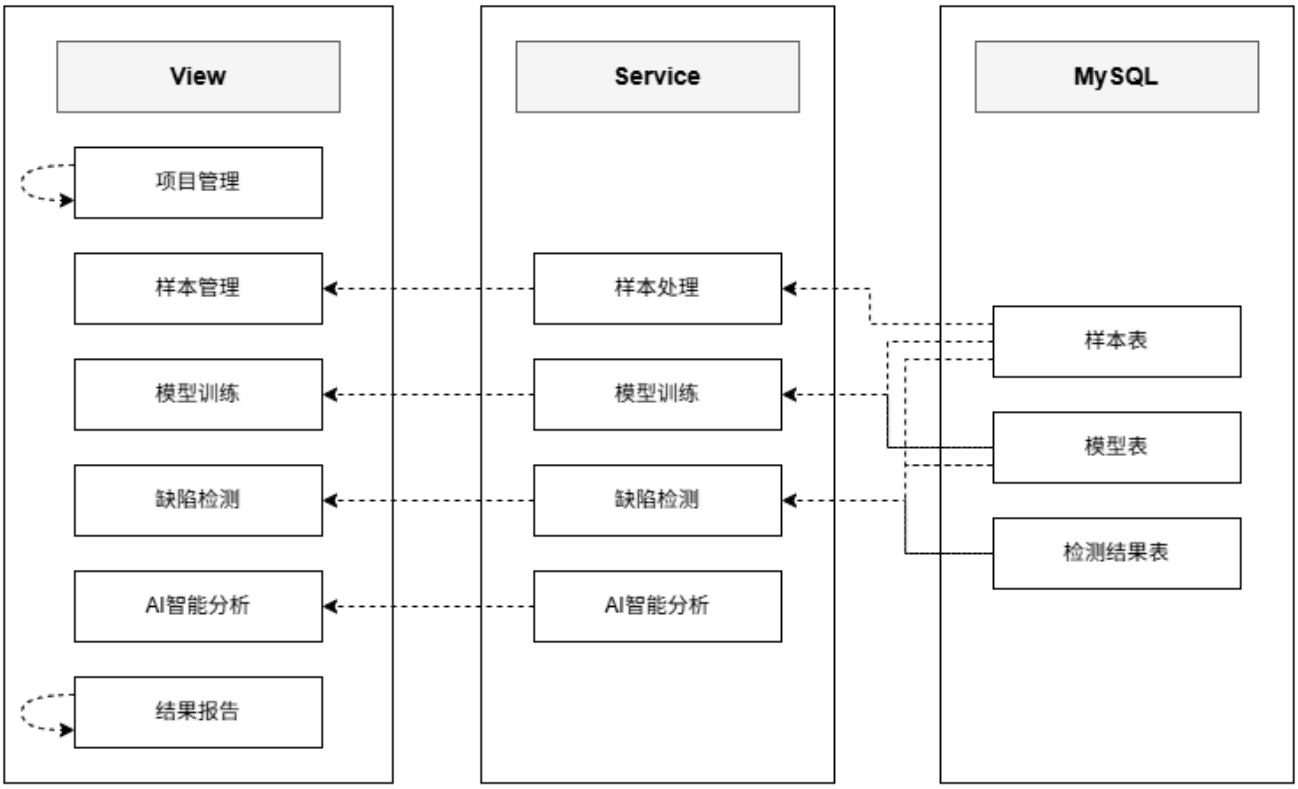
可移植性：客户端应支持跨平台部署，如Windows、Linux、macOS等。

4.3 系统架构设计

4.3.1 总体设计

本系统采用经典的分层体系结构风格，分为展示层、业务逻辑层和数据层三层，展示了整个系统的高层抽象。展示层包含 GUI 界面的实现，主要负责与用户的直接交互，并处理一些简单业务，核心业务逻辑则通过 http 协议与业务逻辑层通信，交由其处理，其中需要持久化的数据交由数据层进行存储和管理。在展示层，使用 Qt Designer 工具设计用户界面的主体部分，然后使用 pyside6 加载相应 ui 文件，实现与用户交互的业务逻辑，这样将 UI 与业务逻辑解耦，便于维护和扩展。业务逻辑层是系统最复杂的部分，负责实现无监督深度学习缺陷检测算法、AI 辅助分析，各模块通过明确定义的 api 接口与展示层通信，接收 http 请求，并返回处理结果。数据层使用 MySQL 数据库，负责业务逻辑层中各类数据的持久化存储、检索和更新。

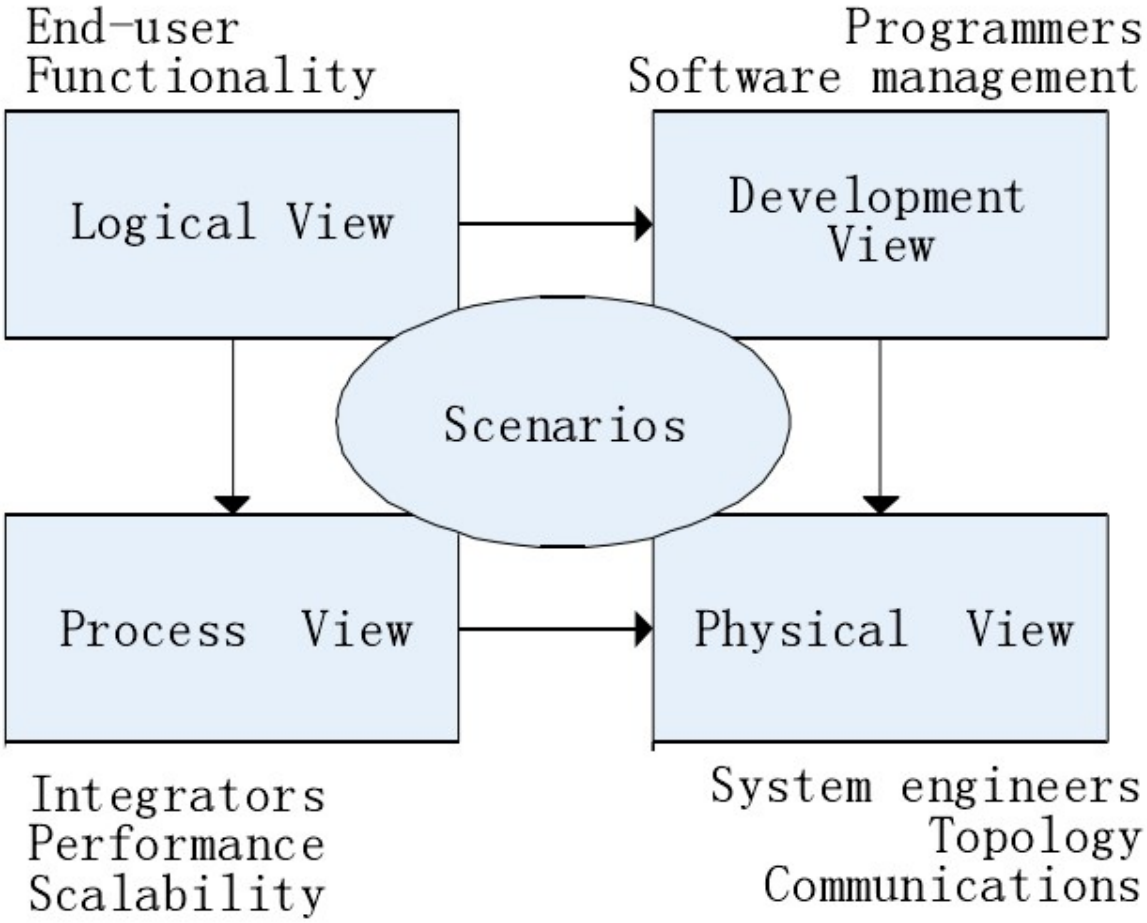
系统总体架构设计图如图 4-2 所示。



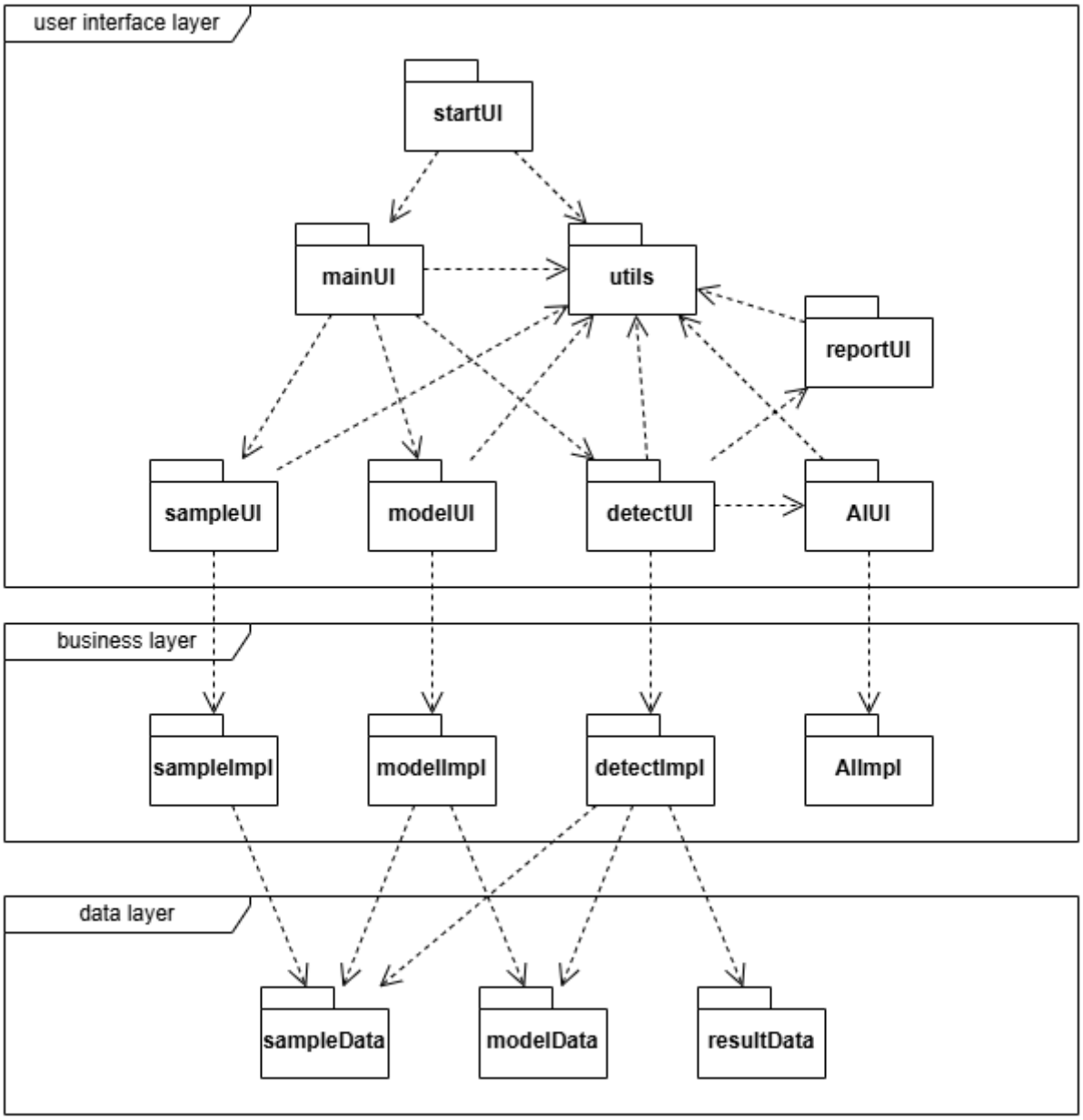
4.3.2 “4+1” 视图

Kruchten在1995年提出了一个 “4+1” 的视图模型。这个模型从 5 个不同的视角，包括逻辑视图、过程视图、物理视图、开发视图和场景视图来描述软件体系结构。每一个视图针对一类人群，关心系统的一个侧面，结合5个视图描述了软件系统结构的全部内容[10]。

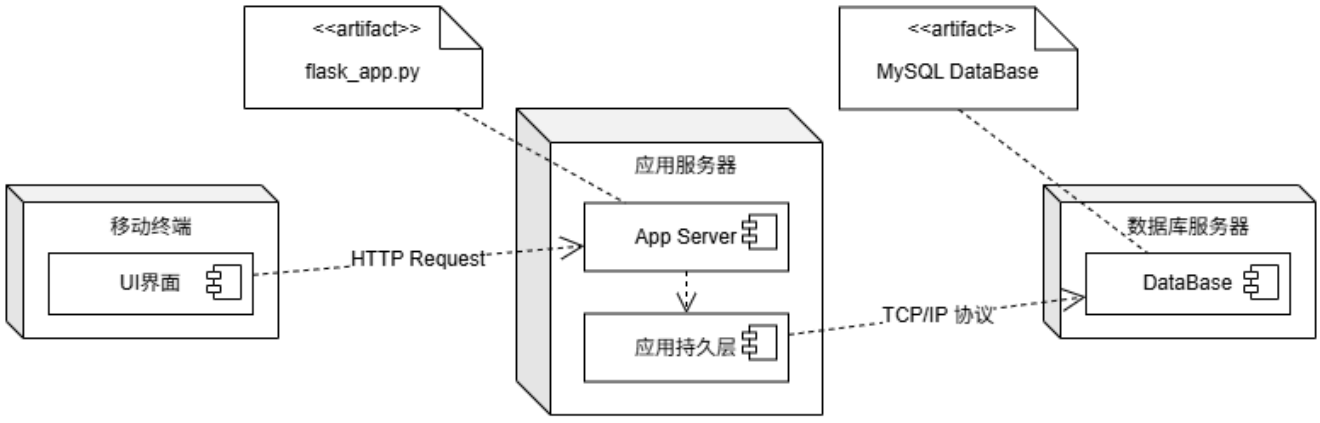
Kruchten 提出的 “4+1” 视图模型如图 4-3 所示。



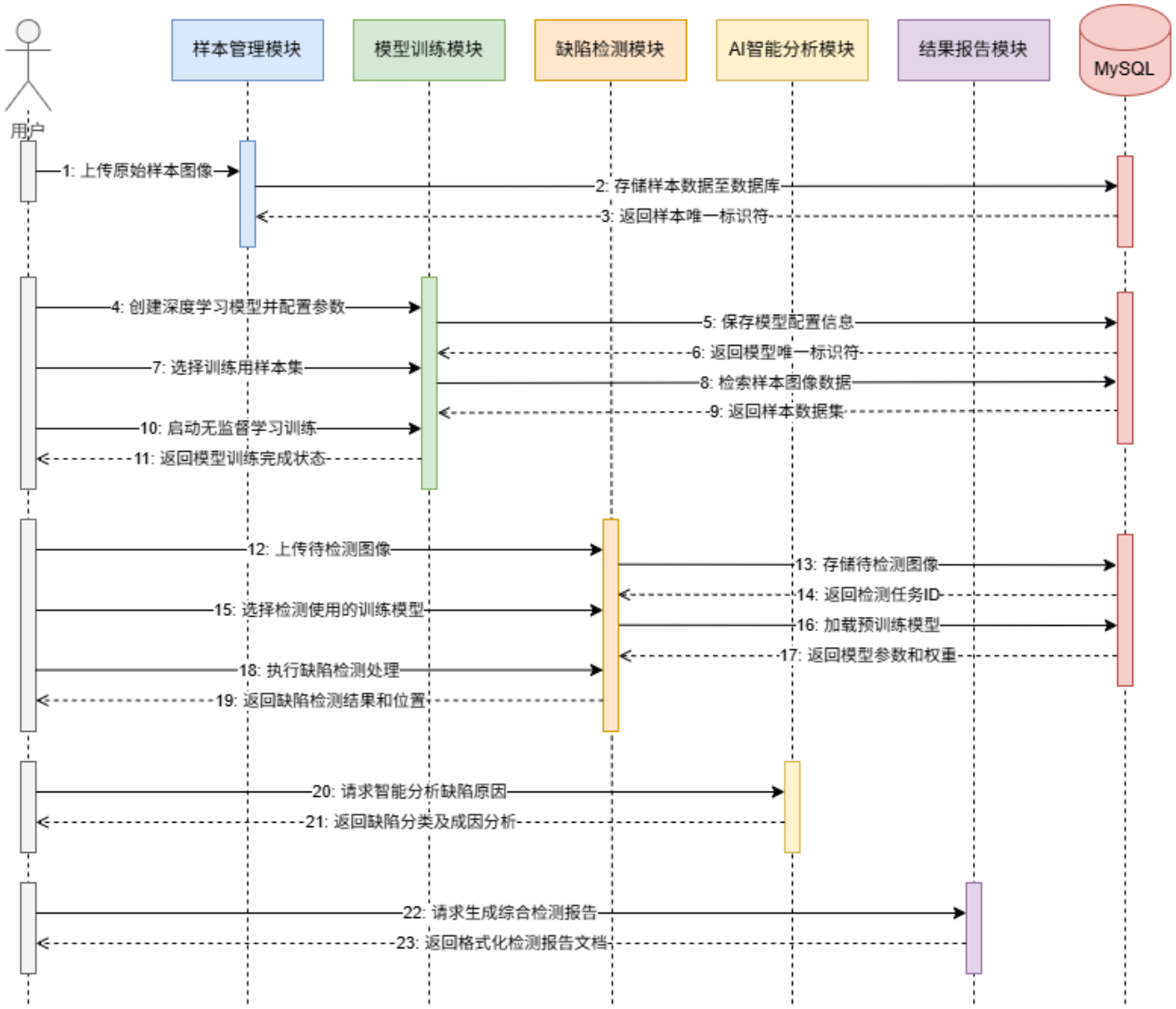
逻辑视图用于从结构化视角描述系统的功能需求，展示了对架构而言重要的元素和它们之间的关系，即系统为用户提供各项服务所具备的静态结构、组件关系和边界约束，反映了系统服务的构建过程。图 4-4 为系统逻辑视图。展示层根据用户行为与业务层进行交互，业务层通过数据层对相关数据进行操作并处理业务逻辑。



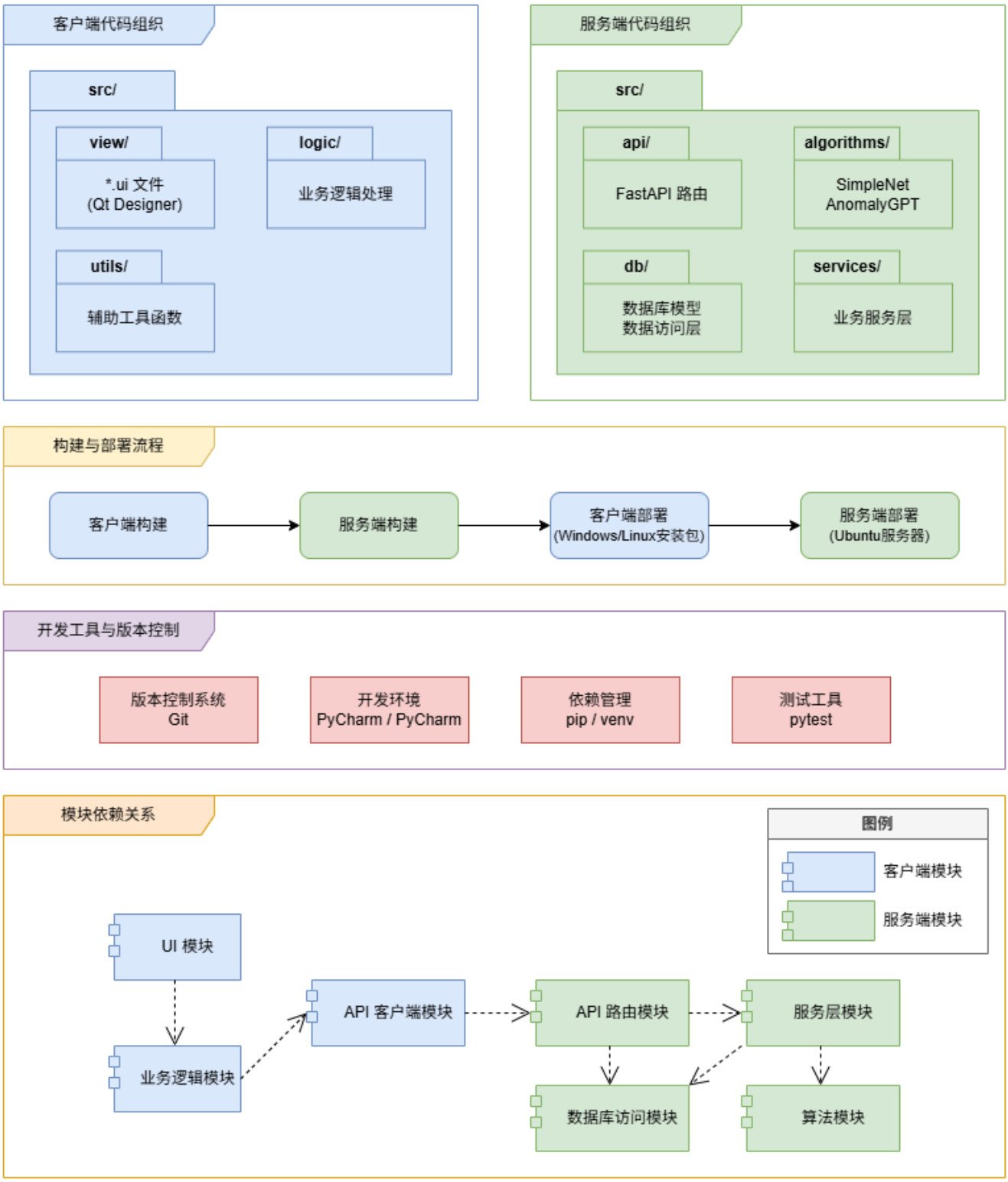
物理视图即部署视图，用于描述系统运行的物理环境或软件环境，前者包括移动终端、服务器等物理设备，后者包括虚拟机等，它们共同展示了系统的主要过程和组件是如何被映射到硬件上的。图 4-5 为系统物理视图。本系统的客户端运行于移动终端上，而服务端以及数据库运行于服务器中。



过程视图即处理视图，用于描述系统的动态行为，即系统组件之间的通信、数据的输入输出，展示了元素之间的并发和交互。过程视图通常由 UML 的顺序图表示，图 4-6 为系统过程视图。



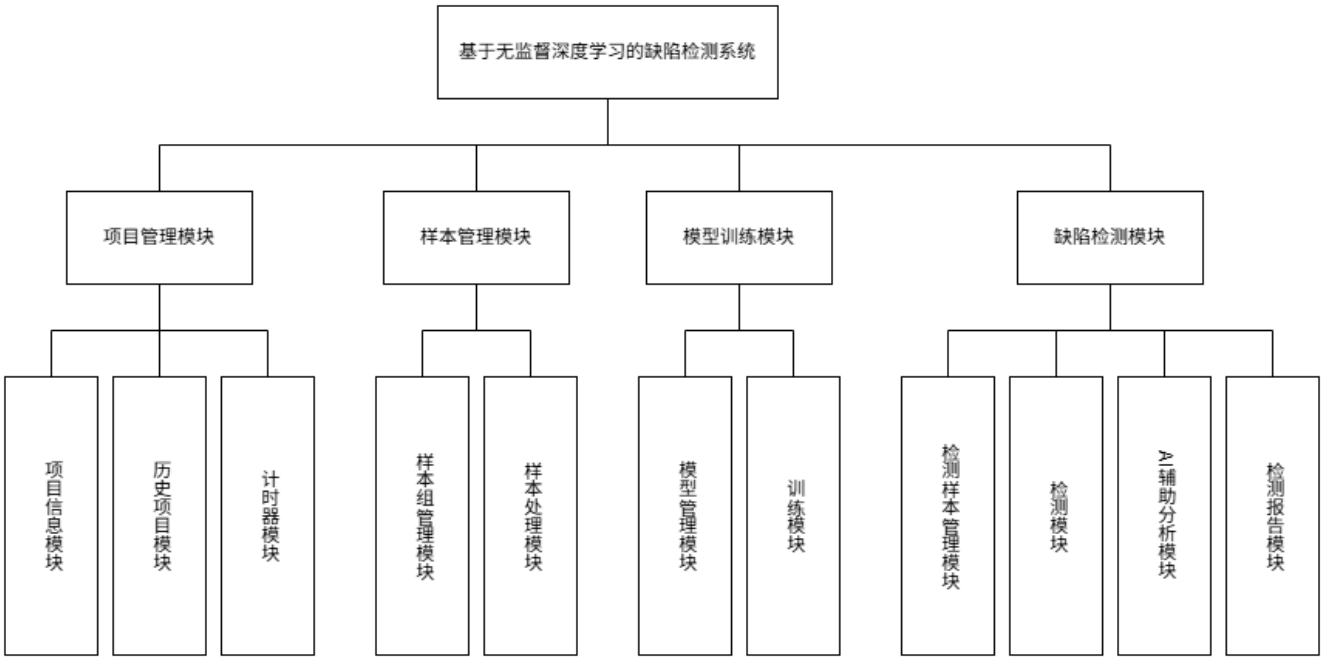
开发视图用于描述软件模块的组织方式和开发管理策略，反映了软件组件的内部组织联系。图 4-7 为系统开发视图，展示了代码组织、构建流程、配置管理和依赖关系。



场景视图即用例图，用于描述参与者与用例间的关系，通过具体的用例场景，展示其他四个视图如何协同工作，反映系统的架构需求和交互设计。图 4-1 为系统场景视图（用例图）。

4.3.2 模块划分

本系统依据功能需求分为项目管理模块、样本管理模块、模型训练模块和缺陷检测模块四个主要功能模块，这四个模块又可以进一步划分为十一个子模块：项目信息模块、历史项目模块、计时器模块、样本组管理模块、样本处理模块、模型管理模块、训练模块、检测样本管理模块、检测模块、AI辅助分析模块、检测报告模块。图 4-7 为系统模块划分图。

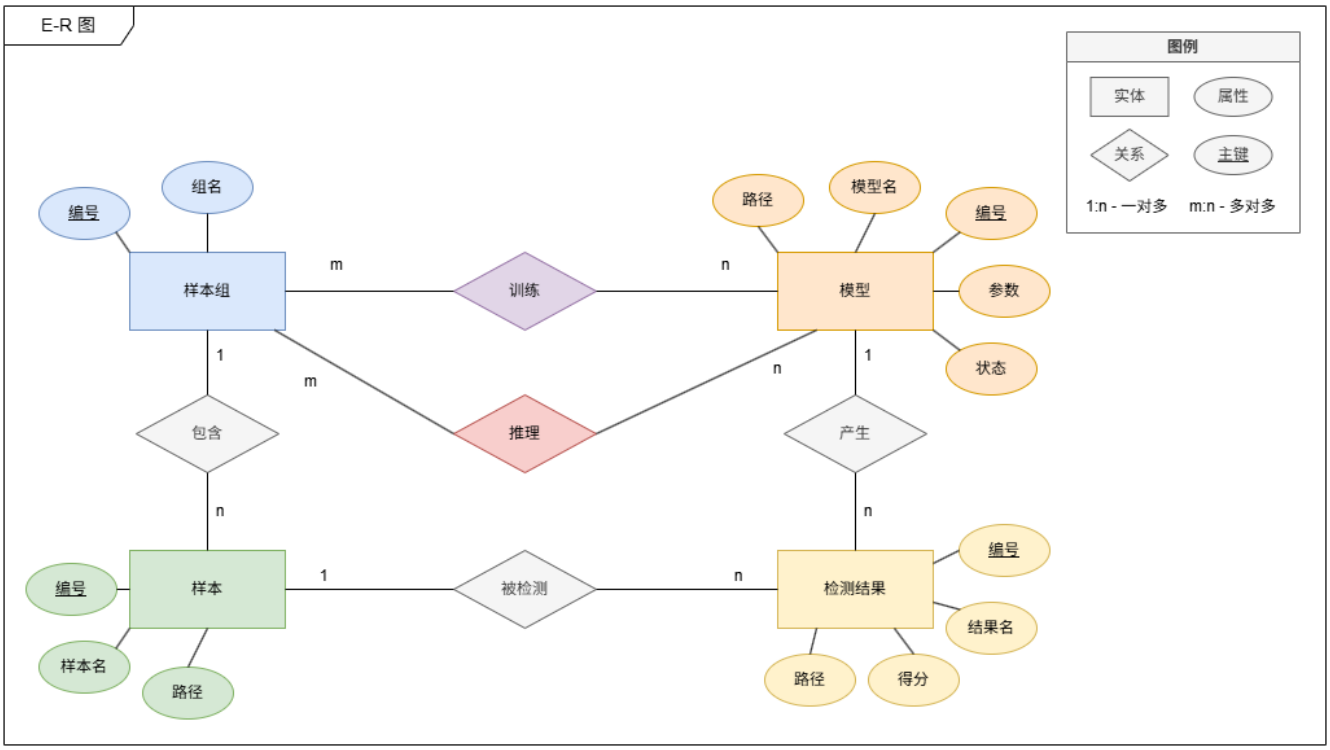


4.4 数据库设计

4.4.1 概述

本系统采用 MySQL 关系型数据库进行数据存储和管理。基于系统功能需求和数据特性，设计了六表结构，包括样本表、样本组表、模型表、检测结果表，以及用于追踪模型训练和推理过程的模型训练样本关联表和模型推理样本关联表。数据库设计遵循实用性和高效性原则，确保数据的完整性和查询效率。

4.4.2 E-R图设计



系统的实体关系图如图 4-8 所示。其中，系统的实体包括：样本组，用于存储样本集合的基本信息；样本，记录单个样本图像的详细信息；模型，存储训练好的模型及其参数；检测结果，记录缺陷检测的结果数据。实体

间的关系包括：一个样本组包含多个样本；检测结果关联到特定的样本和使用的模型；模型与训练样本之间存在多对多关系；模型与推理样本之间存在多对多关系。

4.4.3 核心表结构

1. 样本组表(sample_group)

```
sample_group(
  id INT PRIMARY KEY AUTO_INCREMENT,  // 样本组ID，主键
  name VARCHAR(100) NOT NULL          // 样本组名称
)
```

2. 样本表(sample)

```
sample(
  id INT PRIMARY KEY AUTO_INCREMENT,  // 样本ID，主键
  name VARCHAR(100) NOT NULL,         // 样本名称
  group_id INT NOT NULL,              // 所属样本组ID，外键
  path VARCHAR(255) NOT NULL,         // 样本文件路径
  FOREIGN KEY (group_id) REFERENCES sample_group(id)
)
```

3. 模型表(model)

```
model(
  id INT PRIMARY KEY AUTO_INCREMENT,                    // 模型ID，主键
  name VARCHAR(100) NOT NULL,                           // 模型名称
  path VARCHAR(255) NOT NULL,                           // 模型文件路径
  input_h INT NOT NULL,                                 // 输入高度
  input_w INT NOT NULL,                                 // 输入宽度
  end_acc FLOAT NOT NULL,                               // 结束精度
  layers INT NOT NULL,                                  // 使用的层数
  patchsize INT NOT NULL,                               // 补丁大小
  embed_dimension INT NOT NULL,                         // 嵌入维度
  status ENUM('NEW','TRAINING','READY','INFERRING'),  // 模型状态
)
```

4. 检测结果表(detection_result)

```
detection_result(
  id INT PRIMARY KEY AUTO_INCREMENT,  // 结果ID，主键
  name VARCHAR(100) NOT NULL,         // 结果名称
  sample_id INT NOT NULL,             // 检测的样本ID，外键
  model_id INT NOT NULL,              // 检测的模型ID，外键
  score FLOAT NOT NULL,               // 异常分数
  path VARCHAR(255) NOT NULL,         // 检测结果文件路径
```

```
  FOREIGN KEY (sample_id) REFERENCES sample(id),
  FOREIGN KEY (model_id) REFERENCES model(id)
)
```

5. 模型训练样本表(model_trained_sample)

```
model_training_sample(
  id INT PRIMARY KEY AUTO_INCREMENT,  // 关联ID，主键
  model_id INT NOT NULL,              // 模型ID，外键
  sample_id INT NOT NULL,             // 样本ID，外键
  FOREIGN KEY (model_id) REFERENCES model(model_id),
  FOREIGN KEY (sample_id) REFERENCES sample(sample_id),
  UNIQUE KEY (model_id, sample_id)    // 确保一个样本在一个模型中只被训练一次
)
```

6. 模型推理样本表(model_inferred_sample)

```
model_inference_sample(
  id INT PRIMARY KEY AUTO_INCREMENT,  // 关联ID，主键
  model_id INT NOT NULL,              // 模型ID，外键
  sample_id INT NOT NULL,             // 样本ID，外键
  score FLOAT NOT NULL,               // 异常分数
  FOREIGN KEY (model_id) REFERENCES model(model_id),
  FOREIGN KEY (sample_id) REFERENCES sample(sample_id),
  UNIQUE KEY (model_id, sample_id)    // 确保一个样本在一个模型中只被记录一次
)
```

4.4.4 数据关系与完整性

系统采用外键约束维护表间关系，确保数据的引用完整性：样本表引用 group_id，关联到样本组表；检测结果表、模型训练样本关联表、模型推理样本关联表均引用 sample_id 和 model_id，关联到模型表和样本表

数据库操作采用适当的级联规则，例如：删除样本组时，关联的样本记录自动删除；删除模型时，相关的检测结果仍然保留以便历史查询。

通过这种简洁而功能完备的数据库结构，系统能够有效支持样本管理、模型训练和缺陷检测结果存储，为用户提供完整的数据支持。

第五章 系统实现

5.1 开发环境

客户端在 Windows 11 64 位操作系统上基于 Python 3.11 和 PySide6 开发，客户端运行于 Ubuntu 22.04 服务器上，使用 Git 进行版本控制。

5.2 核心模块实现

5.2.1 项目管理模块

项目管理模块是整个缺陷检测系统的基石，主要负责项目的创建、加载和运行，以及项目信息的管理，为整个缺陷检测系统提供项目层面的数据支持。

5.2.1.1 项目信息模块

项目信息模块是项目管理中的核心，负责项目的创建以及基本信息的存取和管理，为整个缺陷检测系统提供项目层面的数据支持。如图 5-1 所示，用户点击新建按钮创建项目，然后在后续操作中不断完善项目信息；用户点击打开按钮或直接将项目文件夹拖入窗口即可打开已有项目，便于用户继续上一次未完成的工作，或对已完成的工作进行调整后重新处理。

项目信息包括项目元数据，如项目名称、项目路径、模型组、训练样本组、检测样本组等的存取，以及项目相关目录和配置信息。项目结构如图 5-2 所示，整个项目包含项目元数据文件，以及样本组、模型组、检测组三个文件夹，每个文件夹中包含多个子文件夹，里面存放了具体样本组、模型组、检测组的相关信息。其中，项目元数据文件 metadata.json 在新建项目时自动生成，并在后续用户操作中不断更新。而样本列表信息 sample_list.json、模型配置信息 model.json、检测结果列表 detect_list.json 等依次在样本管理、模型训练与缺陷检测过程中被创建或修改。

```
project/                                    # 项目根目录
├── metadata.json                           # 项目元数据文件
├── sample_groups/                          # 样本组文件夹
│   ├── group_1/
│   │   ├── sample_list.json                # 样本列表信息
│   │   ├── *.png                           # 样本图像
│   │   └── ...
│   └── ...
│       └── ...
│
├── model_groups/                           # 模型组文件夹
│   ├── model_1/
│   │   ├── model.json                      # 模型配置信息
│   │   └── trained_samples.json            # 已训练的样本列表
│   └── ...
│       └── ...
│
└── detect_groups/                          # 检测组文件夹
    ├── group_1/
    │   ├── detect_list.json                # 检测样本列表
    │   ├── results/                        # 检测结果
    │   │   ├── *.png                       # 检测结果图像
    │   │   └── ...
    │   ├── reports/                        # 报告文件夹
    │   │   ├── .pdf                        # PDF格式报告
    │   │   ├── *.png                       # 报告图表
    │   │   └── ...
    └── ...
        └── ...
```

5.2.1.2 历史项目模块

历史项目模块主要负责记录、管理和快速访问用户最近操作过的项目。历史项目列表存储在用户主目录下的 .visioCraft/recent_projects.json 文件中，用户新访问的项目会自动添加到列表开头，实现按时间排序；当项目已在列表中时，会先移除旧记录再添加到列表首位。历史项目列表栏详见图 5-1，双击列表项即可打开对应项目，若项目路径不存在，自动从列表中移除并提示用户。

5.2.1.3 计时器模块

计时器是项目中的辅助组件，负责跟踪项目操作时间，帮助用户记录和评估工作时长。该模块基于项目中的 FloatingTimer 类实现，贯穿整个应用程序生命周期。

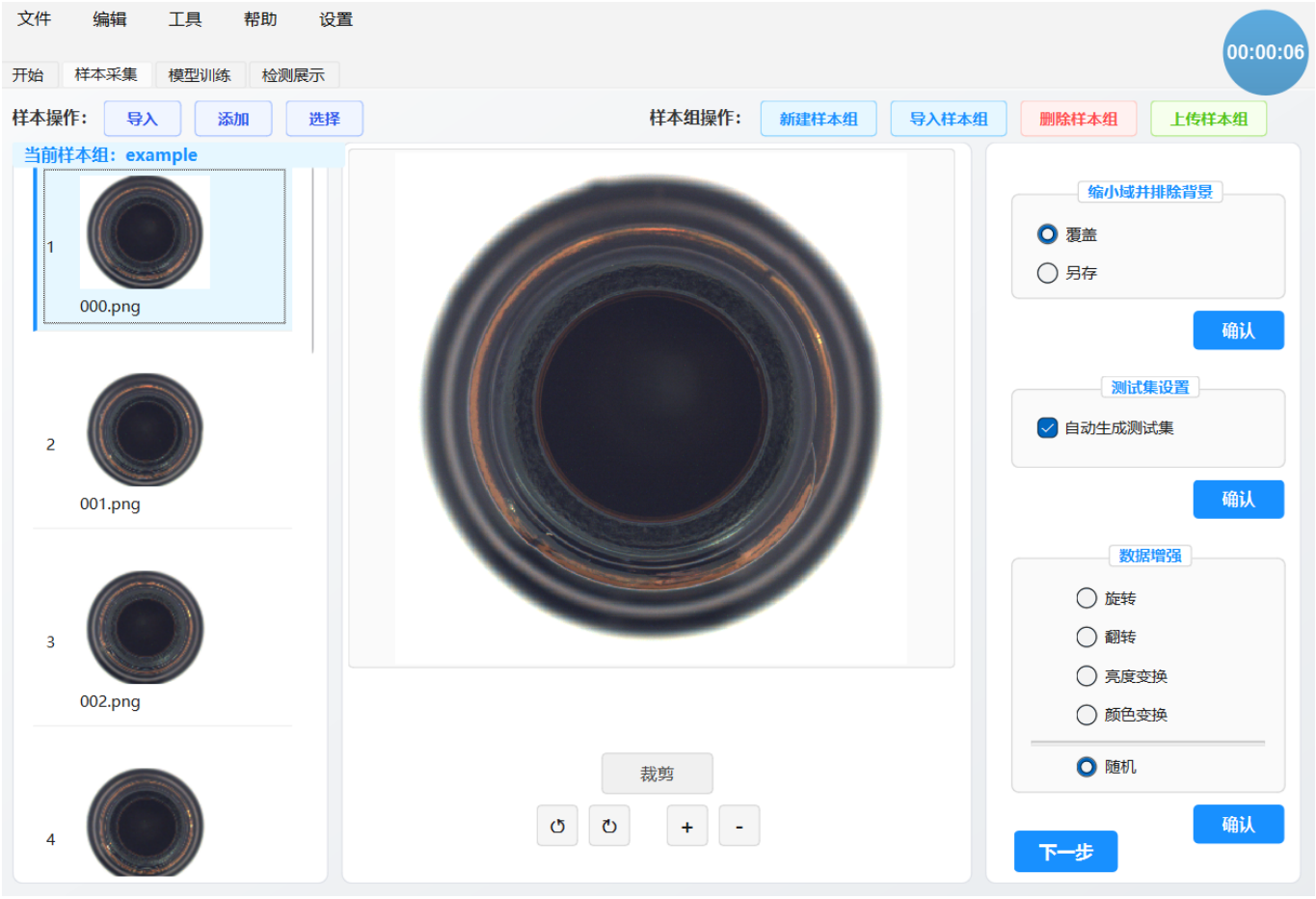
如图 5-1 所示，计时器模块以悬浮球的形式显示，默认位于主窗口右上角，不影响主界面操作，用户也可以拖拽悬浮球到任意位置。悬浮球上显示了当前项目操作的时间，用户可以双击悬浮球来重置计时器，便于分段计时。计时器通过窗口位置监控机制实现了拖拽功能，以及在各界面无缝切换，保证了在整个应用使用过程中时间记录的连续性。

5.2.2 样本管理模块

样本管理模块是缺陷检测系统的基础组件，负责样本的组织、存储和预处理，为后续的模型训练和缺陷检测提供高质量的数据支持。

5.2.2.1 样本组管理模块

样本组管理模块是样本管理的基础，负责组织和管理样本集合及其状态。如图 5-3 所示，用户可以创建、导入、删除，或上传样本组到服务器，其中导入和删除的管理界面详见图 5-4 ，列出了每个已创建的样本组及其状态。样本组通过样本列表栏显示，样本列表同时支持从目录批量导入样本，以及添加单个或多个样本。导入样本后，点击列表项即可切换到对应样本进行查看与编辑，点击选择按钮后可对样本进行多选或删除。当用户新建、切换或删除样本组后，系统会自动更新样本列表栏与界面显示，同时修改样本组的状态。其中，样本组的状态控制了样本列表操作按钮的可见性，仅当样本组存在时才能进行样本的导入、删除等操作。样本编辑完成后，用户可以上传样本组到服务器，以便后续使用。系统会跟踪上传进度，提供动态反馈，当上传完成后，方可进行下一步操作——模型训练。



5.2.2.2 样本处理模块

样本处理模块是样本管理的重头戏，负责单个或多个样本的预处理操作，如裁剪、背景排除与数据增强。裁剪和背景排除的目的是相同的，都是为了去除不必要的图像部分，突出重点区域，从而减少背景干扰，提高检测

精度；数据增强则通过增加样本多样性，提高模型的泛化能力。

背景排除缩小域功能通过智能分割算法识别并保留图像的主体部分，确保后续数据增强以及模型训练集中于目标区域，适用于有明显背景的样本。相比手动裁剪，它可自动化对样本进行批量处理，提高了处理效率。使用 OpenCV 库进行图像阈值处理、轮廓检测和过滤，基于形态学操作进行轮廓优化，消除噪点和细小碎片。具体来说，就是先把图像灰度转换和二值化，以分离前景和背景，然后进行轮廓检测和筛选，保留主要目标区域，最后自动计算目标区域的最小外接矩形，并裁剪图像。另外，为了避免过度裁剪，系统会自动添加适当边距，确保目标的完整性。

图像裁剪功能使用户能够通过交互式裁剪框手动去除样本中无关区域，可以进一步优化背景排除的效果，适用于背景排除算法效果不佳且样本量较少的情况。基于自定义的 ResizableRectItem 类实现可调整大小的裁剪框，在 QGraphicsView 中通过 QGraphicsScene 构建互动场景，实现矩形的拖拽和缩放。用户点击裁剪按钮后，裁剪框会自动显示在图像上，用户将鼠标移至裁剪框的边界点时，会自动切换光标样式，此时可拖拽边框来调整裁剪区域的大小。缩放或移动矩形区域后，裁剪框以外的图像部分会自动变模糊，反差对比明显。点击接受按钮，裁剪框自动消失，且裁剪区域以外的部分被丢弃。此时，用户可选择保存裁剪图像，样本列表栏中图像的预览图以及中央的大图会自动刷新并自适应。在以上过程中，用户可以随时点击取消按钮，取消裁剪操作，此时裁剪框会自动消失，图像恢复原状。另外，系统在裁剪时会临时禁用其它编辑功能，以防用户误操作。

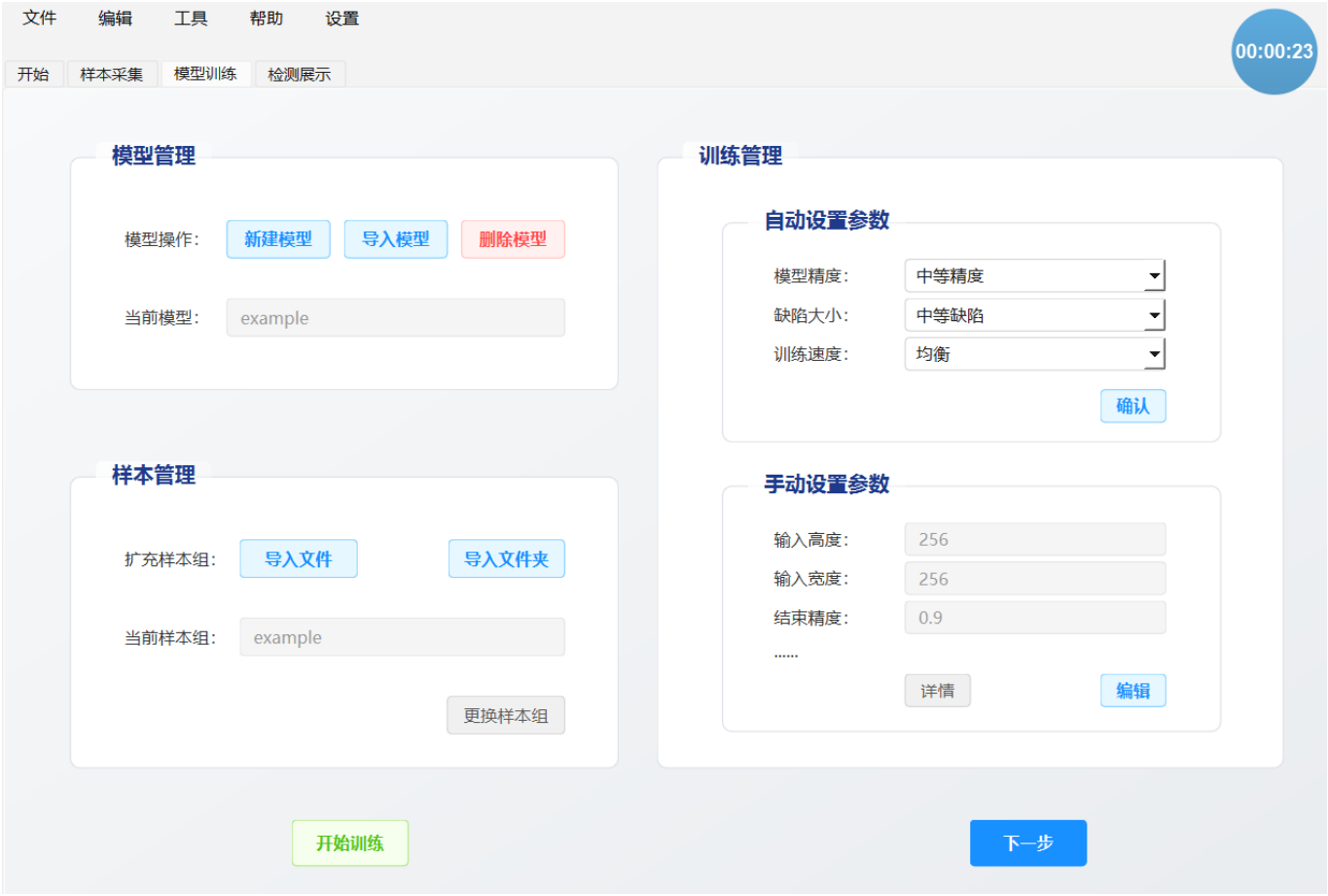
数据增强功能提供了多种图像变换方式，能扩充训练样本集，增强模型的泛化能力，适用于样本量较少的情况。数据增强支持批量处理，使用 OpenCV 库对选中的样本进行几何变换和外观变换，包括旋转、翻转、亮度变化和颜色变换。用户既能选定特定方式，也可以选择随机变换，系统会自动生成变换后的图像，刷新后显示在样本列表栏中。另外，由于无监督学习模型的训练集要求是正向样本，数据增强功能只会对样本进行轻微调整。

5.2.3 模型训练模块

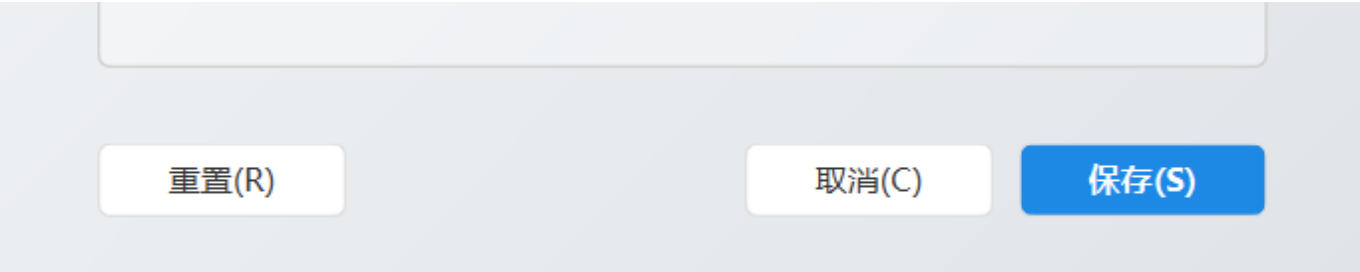
模型训练模块是应用基于无监督深度学习的缺陷检测算法的核心组件，负责管理和训练缺陷检测模型。

5.2.3.1 模型管理模块

模型管理模块是模型训练的准备工作，负责组织和管理模型及其配置信息。如图 5-5 所示，用户可以创建、导入、删除模型，以及调整模型参数，其中导入和删除的管理界面与样本管理模块类似，唯一的区别就是用灰色、黄色和绿色分别标识了未训练、训练或推理中、已训练这几种状态，此处不再赘述。当用户新建或导入模型后，可进行参数设置，系统提供了参数映射功能，通过模型精度、缺陷大小、训练速度三个选项的配置可自动生成模型参数建议值，为新手或想要快速完成训练过程的用户给予方便。该机制首先从精度维度确定特征提取深度和嵌入维度，再从缺陷大小维度设定输入分辨率和补丁大小，最后通过训练速度维度对参数进行整体微调，实现参数间的协同配置。当然，专业用户也可以在详细编辑界面进行手动设置，如图 5-6 所示，系统暂时只支持配置输入尺寸、结束精度、补丁大小、特征层、嵌入维度这几个参数。参数设置完成后，模型相关信息会被自动同步到服务器，同时保存到本地，以便后续使用。

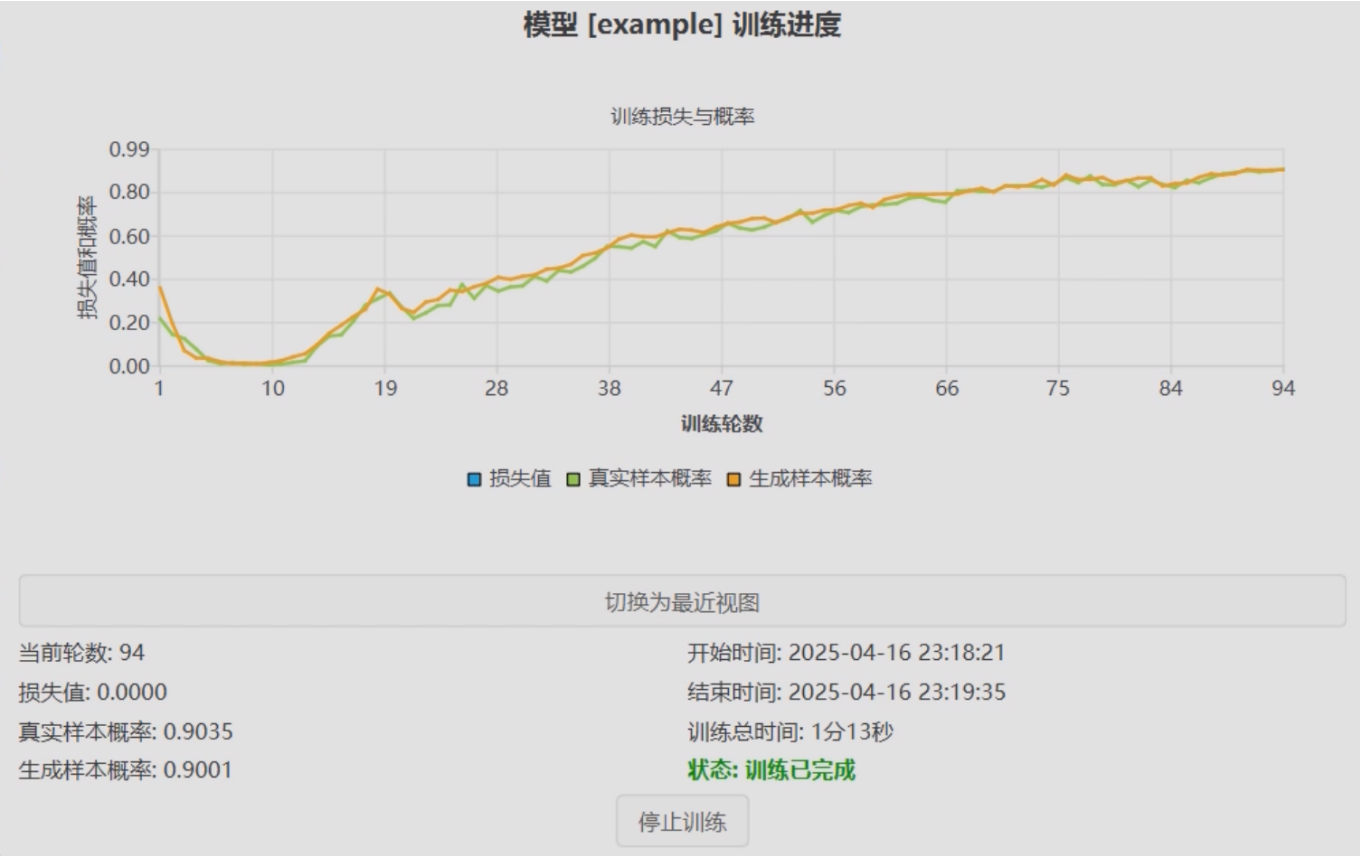

文件
编辑
工具
帮助
设置
00:00:23
开始
样本采集
模型训练
检测展示
模型管理
模型操作:
新建模型
导入模型
删除模型
当前模型:
example
样本管理
扩充样本组:
导入文件
导入文件夹
当前样本组:
example
更换样本组
训练管理
自动设置参数
模型精度:
中等精度
缺陷大小:
中等缺陷
训练速度:
均衡
确认
手动设置参数
输入高度:
256
输入宽度:
256
结束精度:
0.9
.....
详情
编辑
开始训练
下一步


缺陷检测模型参数设置
基本参数
输入尺寸:
256
256
补丁大小:
5
结束精度:
0.92
特征提取参数
嵌入维度:
512
特征层:
layer1
layer2
layer3
layer4
提示：layer1为浅层特征，layer4为深层特征，请至少选择1个特征层



5.2.3.2 训练模块

训练模块负责模型的实际训练过程，包括训练执行、训练进度可视化以及模型状态控制。训练前系统会自动验证样本组和模型组的有效性，比如样本组是否为空、模型是否处于空闲状态等。启动训练后，进入等待态，同时界面显示加载动画，直至获取到第一轮训练结果。此时开始显示实时训练进度，如图 5-7 所示，显示了动态概率曲线，同时记录了当前训练轮次、训练时间、训练精度等指标。其中训练曲线可视化组件采用 Qt 图表框架构建，定时从服务器拉取训练数据并实时更新图表，并通过 QChart 和 QLineSeries 在坐标轴上实现动态绘制。同时，该组件支持全局视图与局部放大图的切换，便于用户灵活观察整体趋势与细节变化。另外，如果用户急需使用模型，可以随时点击停止训练按钮，终止训练过程。当模型达到结束精度或用户手动终止训练后，模型训练完成，自动恢复空闲状态，此时用户可进行下一步操作——缺陷检测。



5.2.4 缺陷检测模块

缺陷检测模块是整个缺陷检测系统的核心应用环节，负责对检测样本进行缺陷识别、可视化、AI辅助分析，最后生成统计报告。

5.2.4.1 检测样本管理模块

检测样本管理模块与样本管理模块类似，负责组织和管理检测样本集合及其状态。

5.2.4.2 检测模块

5.2.4.3 AI辅助分析模块

5.2.4.4 检测报告模块

5.3 接口与其他模块

5.3.1 API 接口

5.3.2 其他模块

LoadImages、LoadingAnimation、UploadSampleGroup_HTTP

5.4 部署

5.4.1 服务器端部署

服务器由哈工大多模态智能及应用研究中心支持，项目运行于 ubuntu 22.04 环境，将服务器端打包为镜像后可快速部署于任意服务器上运行。

5.4.2 客户端部署

客户端通过 PyInstaller 打包为可执行文件，包含源代码、ui 文件，以及 PySide6 等库文件，可部署在 Windows 11 64 位操作系统上运行。

5.5 界面展示

关键功能操作流程图、界面截图

第六章 测试与分析

6.1 数据集

6.2 实验设计

三组样本：正常、小缺陷、大缺陷 -> 通过率

两组样本：速度优先、精度优先 -> 时间与准确率

6.3 结果分析

第七章 总结与展望

7.1 工作总结

本研究以工业质检场景中的实际需求为导向，设计并实现了一套基于无监督学习的缺陷检测系统。针对传统方法依赖人工标注数据、对复杂环境适应性不足等问题，提出了多尺度特征融合、动态参数调整等核心算法，并结合工程化优化策略，实现了高效、低成本的缺陷检测方案。系统通过仅使用正常样本训练模型，显著降低数据准备成本；通过客户端-服务器架构与轻量化设计，适配不同硬件环境；通过可视化交互界面，简化操作流程，为非专业人员提供便捷的使用体验。实验表明，系统在复杂工业场景中表现出稳定的检测能力，能够有效替代传统人工检测，为中小企业的智能化转型提供了可行的技术路径。

7.2 未来展望

尽管本系统在基础功能上达到预期目标，但仍存在进一步优化的空间。未来工作可从以下方向展开：

1. **功能扩展**：结合少量标注数据，引入缺陷分类模块，实现缺陷类型的自动识别与分级；
2. **场景适应性增强**：针对低对比度材质（如玻璃、塑料）的检测难题，探索多模态数据融合方法（如结合光谱或深度信息）；
3. **动态学习能力**：开发在线学习机制，使系统能够根据新增数据自动更新模型，适应产线工艺的持续变化；
4. **边缘计算优化**：针对低成本硬件（如嵌入式设备），进一步压缩模型体积，提升计算效率，拓展适用场景；
5. **用户交互改进**：集成自动化参数调优功能，减少人工干预，提升系统的智能化水平。

本研究验证了无监督学习技术在工业质检领域的潜力，未来将通过持续优化与功能迭代，推动其在更广泛制造场景中的应用，助力工业智能化发展。