四川大學

SICHUAN UNIVERSITY

基于生成式 AI 的个性化文创图像作品设计系统

项　目　总　结　报　告

任课教师　杨波

学　　院　计算机学院

专　　业　计算机科学与技术

组　　别　第一组

组　　长　郑仕博

成　　员　陈奕嘉，苏泳豪

2025 年 7 月 2 日

目录

1 个人报告

1.1 项目基本情况

本项目名为“基于生成式 AI 的个性化文创图像作品设计系统”，旨在响应当前文创市场对于个性化、创新性作品的迫切需求，解决现有文创产品同质化严重、难以满足用户多样化表达的痛点。受到国家推动文旅深度融合政策及《如果国宝会说话》等优秀文化传播案例的启发，我们团队期望通过前沿的人工智能技术，特别是生成式模型，来“激活”沉睡的文化遗产，赋予普通用户便捷、高效的创作能力，使其能够轻松地将个人创意与文化元素结合，设计出独一无二的文创图像作品。

项目的核心技术路径围绕先进的生成式 AI 模型展开。主要开发工作分为三大模块：首先是模型微调与优化，我们选用业界表现优异的 AnyText 模型为基础，针对中文汉字的渲染特性进行专门的优化训练，并结合 stable diffusion v1-5 模型，通过 Dreambooth 方法进行微调，使其能生成具有特定文创风格的图像；其次是数据集的构建与处理，我们制作了两份关键数据集，一份是基于 AnyWord-3M 数据集筛选、清洗后得到的约 40 万条高质量图文对，用于增强模型的文字控制能力，另一份是通过网络爬虫技术采集的约 1000 张聚焦中华传统文化与文物的图片，用于模型的风格化微调；最后是系统的实现与部署，我们基于 Gradio 框架搭建了功能完善、操作直观的交互式 Web 界面，成功实现了“文本到图像生成”和“图像内文字编辑”两大核心功能，并最终通过 Docker 技术将整个项目封装，部署在 ModelScope 等云端平台，实现了项目的完整交付。

本项目由四川大学计算机学院的杨波老师悉心指导。我们的团队“第一组”由三名成员组成，分别是担任组长的郑仕博、组员陈奕嘉以及我本人。在长达近四个月的项目周期里，我们从最初的立项策划、需求分析，到中期的技术攻坚、代码实现，再到后期的系统部署、测试与文档完善以及软著申请，团队成员紧密协作，共同经历并克服了多次技术迭代和难题挑战，最终圆满完成了项目的预定目标。

1.2 承担的主要工作情况

在本项目中，我主要承担了前端后端框架的完整开发与实现、项目所需数据集的搜集与标注处理、辅助模型进行调试，以及为项目的最终部署提供技术支持等关键职责。我的工作贯穿了项目始终，是连接底层 AI 模型能力与上层用户体验的桥梁。以下是我在项目不同阶段，按照时间顺序详细阐述的具体工作内容：

项目启动与框架搭建阶段 (Week 4-5) 在项目初期，我的首要任务是为整个系统搭建一个稳定且可扩展的前后端框架。第四周，我通过在 GitHub 等开源社区进行广泛调研，寻找到了多个适合本项目需求的 Web UI 框架模板。经过对这些模板的代码结构、技术栈和可定制性进行细致评估后，我选择了一个基于 Gradio 的模板作为基础，并对其进行了大量的修改和定制化开发，使其符合我们项目的特定功能布局和交互逻辑，从而迅速确定了整个应用的基本框架。这为后续功能的快速迭代开发奠定了坚实的基础。进入第五周，随着模型训练对数据的需求日益明确，我开始

承担数据集的收集与整理工作。我负责统筹和整合所有团队成员搜集到的数据资源，建立了一套初步的数据管理流程，并继续改进前后端的代码，为后续与模型的对接做准备。

数据处理与前端功能深化阶段 (Week 6-8) 从第六周开始，我的工作重心进一步向数据处理和前端功能实现倾斜。我继续收集适合模型训练的文创主题数据集，并承担了一项关键的数据标注任务。我设计并编写了自动化脚本，对 wd14-convnextv2-v2 数据集的标签进行处理，以满足我们模型的训练要求。在第七周，为了进一步提升数据质量，我独立编写了名为 `revise_caption.py` 的 Python 脚本，该脚本能够精准地对数据集中的标签进行批量修改，例如去除无用的“噪声”提示词、规范标签格式等，这为后续模型训练的收敛和效果提升起到了至关重要的作用。与此同时，我持续推进前后端代码的编写工作。第八周，在完成所有数据收集任务后，我与组长一同参与了模型的初步训练，从旁辅助调试，确保模型能够顺利运行。

核心功能实现与系统整合阶段 (Week 9-10) 第九周是项目实现的关键节点。在这一周，我全面完成了前后端所有核心功能的代码编写工作。我成功实现了系统的两大核心功能模块：“文本生成”和“图像编辑”。在“文本生成”模块中，用户可以通过输入描述性的提示词，并用双引号指定需要渲染到图像上的文字，然后在画布上通过手绘、拖拽矩形框或随机生成的方式指定文字位置，最终生成文创产品。在“图像编辑”模块中，用户可以上传一张自己的图片，使用画笔涂抹需要修改的区域，并输入新的提示词进行局部重绘。此外，我还开发了详细的参数修改界面，用户可以自由调节 CFG-Scale、eta、Strength 等超参数，以生成更具个性化的作品。这一系列工作的完成，标志着我们的系统从概念走向了可实际操作的产品。第十周，为了提升项目文档的专业性和规范性，我主动承担了使用 LaTeX 重新排版《需求分析》文档的工作，使其格式更加美观、结构更加清晰。

项目部署与交付准备阶段 (Week 11-14) 进入项目后期，我的工作重点转向了项目的封装、部署与测试。第十一周，我参与了项目 Dockerfile 的编写工作。在将项目容器化的过程中，我们遇到了棘手的网络问题，导致在 Docker build 过程中部分依赖包无法下载。面对这个难题，我与团队成员共同研究，最终通过配置虚拟网卡的方式成功解决了此问题，确保了镜像的顺利构建，并成功将项目封装并上传至 Docker Hub。第十二周，为了方便在 ModelScope 等平台进行部署，我编写了项目的 requirements.txt 文件，精确地列出了所有 Python 依赖，并与组员一起将整个项目成功部署到了 ModelScope 平台。在最后的两周 (第十三、十四周)，我对项目产品进行了全面的功能和压力测试，验证各项功能的稳定性和用户体验，并记录了测试中发现的问题。同时，我积极参与了最终汇报 PPT 的制作和所有项目总结文档的修改与完善工作，为项目的最终展示和交付做出了贡献。

1.3 项目实践过程中遇到的问题及处理结果

在整个项目从零到一的开发过程中，我们团队不可避免地遇到了一系列技术和实践上的挑战。作为前后端开发和部署的相关负责人，我直面并解决了很多具体问

题。

Week 4-6: 前后端框架搭建与技术熟练度问题在项目初期的第四、第五周, 我的任务是搭建前后端基本框架并使其能与 AI 模型对接。但到第六周时, 问题逐渐显现: 由于我对 Gradio 框架以及一些复杂的前端交互逻辑的实现不够熟练, 导致代码中存在较多 Bug, 功能运行不稳定, 无法高效地为模型调试提供支持。面对这个问题, 我采取了“学习与实践并行”的策略。我投入了大量时间阅读 Gradio 的官方文档, 并在网上查找相关的教程和开源项目案例进行学习。同时, 我积极与小组同学讨论, 他们从模型和用户需求的角度为我提供了很多有价值的建议。经过不懈的努力, 我逐渐掌握了 Gradio 的开发模式, 修复了大部分已知 Bug, 保证了前后端框架的稳定性, 为后续核心功能的顺利实现打下了基础。

textbfWeek 7-8: 数据处理与辅助模型调试的挑战 第七周, 团队的核心瓶颈在于模型效果有待提高。我编写的数据清洗脚本（`revise_caption.py`）的效率和准确性直接影响着训练数据的质量, 进而影响模型效果。因此如何正确处理数据, 并为模型提供一个能够快速验证效果的前端界面, 成为了我面临的挑战。到第八周, 前后端开发进入收尾阶段, 问题则转变为如何确保所有功能接口与最终模型完美兼容。我对自己编写的数据处理脚本进行了多次测试和重构, 确保其能够稳定、准确地处理大规模数据集。同时, 我加速了前端功能的开发, 提前实现了图片上传、参数调整等功能, 为模型团队提供了一个可以即时反馈的调试环境。在第八周的收尾工作中, 我与组长紧密沟通, 根据模型最终确定的输入输出格式, 对前后端的接口进行了最后的调整和测试, 确保了系统的顺利整合。

Week 9-10: 核心功能完成后的界面美化与部署准备第九周, 虽然系统核心功能已全面实现, 但 Web 界面的整体布局、色彩搭配和交互细节还比较粗糙, 美观度不足, 距离一个成熟的产品还有差距。此外, 第十周我们开始规划部署方案, 但当时团队对于如何将这样一个包含大型模型的项目进行容器化封装 (Docker) 还没有成熟的方案。针对界面美化问题, 我认识到用户体验的重要性。我利用 Gradio 提供的主题和自定义 CSS 功能, 对界面进行了初步的美化, 调整了组件间距、字体大小和颜色, 使界面看起来更加协调。同时, 我也优化了一些交互流程, 比如在参数设置区增加了更详细的说明文字。对于部署问题, 我开始研究 Docker 技术, 学习 Dockerfile 的编写规范, 并着手准备将项目封装, 为下一阶段的部署工作扫清了前期的知识障碍。

Week 11: Docker 容器化中的网络瓶颈问题在第十一周正式编写 Dockerfile, 准备将项目封装为镜像时, 我们遇到了一个困难的问题。在执行 docker build 命令时, 由于网络环境的限制, 一些国外的 Python 依赖包和系统库下载极其缓慢, 甚至频繁超时失败, 导致镜像始终无法构建成功。这个问题一度让我们的部署工作陷入停滞。起初我们尝试更换国内的 pip 镜像源和 apt 源, 但效果不佳, 部分核心依赖仍然无法获取。在查阅了大量资料并进行多种尝试后, 我们发现根本原因在于构建环境无法直接访问特定的外部网络。最终, 我与团队成员协作, 采用了配置虚拟网卡的技术方案。我们通过在宿主机上设置代理, 并在 Dockerfile 中配置相应的环境变量, 使得 Docker 在构建过程中能够通过该虚拟网卡访问外部网络。这个方案完美地解决了

依赖下载问题,我们成功地构建了项目镜像,并将其顺利推送到了 Docker Hub,这是我们项目能够实现便捷部署的关键一步。

Week 12: ModelScope 云端部署的性能挑战第十二周,我们将项目成功部署到 ModelScope 社区后,虽然实现了在线演示的目标,但很快发现了一个新的问题:由于 ModelScope 平台的机制,应用实例无法持久化存储。每次有新用户访问或长时间无操作后,实例会被回收,下一次访问时需要经历一个完整的“冷启动”过程,包括重新下载模型、加载环境等,整个过程非常缓慢,严重影响了用户体验。我们分析发现这个问题与项目设置为“公开”有关,并且是平台本身的特性,难以从代码层面根本解决。面对这个客观限制,我们采取了多方面的补救措施。首先,我们在项目的使用说明中明确告知用户可能存在的启动缓慢问题,以管理用户预期。其次,我们讨论并探索了设置“预热”机制的可能性。虽然最终未能完美实现,但这个过程也让我们对云原生应用部署的性能优化有了更深的理解。

Week 13-14: 系统终期测试与文档完善在项目最后的两周,主要问题不再是技术攻坚,而是如何确保产品的质量。我们需要拟定最终的测试思路,对整个系统进行全面、细致的测试,并对所有项目文档进行最后的修改和完善,确保没有遗漏和错误。我承担了产品的主要测试工作,从用户角度对文本生成、图像编辑的每一个功能点和参数组合进行了测试,记录并修复了一些遗留的细节 Bug。同时,我与其他组员分工合作,交叉审阅了所有的项目文档,对其中的内容进行了补充和润色,并参与了最终汇报 PPT 的制作,确保了项目最终以高质量的状态完成交付。

1.4 参与项目过程的体会与自我评价

回顾这四个月的项目历程,我深感收获巨大。这不仅是一次将理论知识应用于实践的宝贵机会,更是一场关于技术、协作和个人成长的全面锻炼。

1.4.1 参与项目过程的体会

首先,我深刻体会到“从无到有”的创造性满足感。作为项目前后端框架的主要构建者,我亲手将一个最初仅停留在概念阶段的想法,一步步地通过代码变成了用户可以真实交互、可以产生价值的在线应用。从搭建第一个空白页面,到实现复杂的图文生成逻辑,再到解决棘手的部署难题,每一个环节的突破都给我带来了巨大的成就感。

其次,我认识到理论与实践之间的鸿沟以及填补它的重要性。之前学到的编程知识、开发知识,在实际项目中会以各种意想不到的形式接受考验。例如,Docker 和 Gradio 这些工具,虽然之前有所耳闻,但只有在真正需要用它们来解决部署和快速原型开发问题时,我才开始深入学习并掌握它们。特别是解决 Docker 网络问题的经历,让我体会到,面对复杂工程问题时,扎实的计算机网络知识、强大的信息检索能力和敢于尝试的动手精神缺一不可。

再者,我领悟了团队协作的重要性。这个项目涉及模型、数据、前后端、部署等多个环节,单凭一人之力很难完成。在项目中,我与同组同学分工明确,紧密配合。

当我被前端交互逻辑困扰时，他们能从用户和模型的角度给我启发；当他们需要将模型能力展示给用户时，我构建的前后端界面则提供了最好的载体。我们每周的例会和日常的沟通，保证了信息的通畅和目标的统一，使得我们能够作为一个有机的整体，高效地协同作战，共同克服难关。

1.4.2 自我评价

在技术能力与贡献方面，我作为前后端开发的负责人，负责了从技术选型、框架搭建到核心功能实现等一系列工作，并最终交付了一个功能完备、运行稳定的 Web 应用。我不仅熟练运用了 Python 和 Gradio，还通过编写数据处理脚本，直接支持了上游的模型训练任务。在项目部署环节，依靠技术攻坚与团队协作的能力，成功解决了 Docker 网络限制这一关键瓶颈，还完成了 requirements.txt 的编写和 ModelScope 的部署工作。此外，我主动学习并使用 LaTeX 重构技术文档，体现了我对项目规范化和高质量交付的追求。

在问题解决与创新思维方面，我能够积极主动地面对开发过程中遇到的各种挑战。无论是技术生疏带来的编码困难，还是部署环境中出现的意外状况，我都能沉下心来分析问题，并通过查阅资料、动手实验等方式寻找解决方案。在解决 Docker 网络问题时，没有拘泥于常规方法，而是与团队共同探索并成功实践了更具创新性的解决方案，这体现了我良好的问题解决能力和一定的创新意识。

当然，我也清醒地认识到自身的不足。例如，在 UI/UX 设计方面的专业知识还有所欠缺，导致产品界面的美观度和用户体验还有很大的提升空间。此外，对于大规模 Web 应用的后端性能优化、高并发处理等方面的知识储备尚浅。这些都将是我未来学习和努力的方向。

这次项目经历是一次意义非凡的淬炼。我不仅将所学知识融会贯通，更在实战中锻炼了工程能力、协作能力和解决复杂问题的能力，为我未来的学习和职业生涯奠定了坚实的基础。

2 项目总结报告

2.1 绪论

2.1.1 项目背景

本作品的核心创意来源于当前市场上文创产品同质化严重，难以满足游客日益增长的个性化需求的痛点。习近平总书记关于推动文化和旅游融合发展，将文化旅游业培育成为支柱产业的指示，以及《如果国宝会说话》等成功案例，激发了通过创新方式“激活”文化遗产，赋能个体创造独特文创作品的想法。本项目旨在解决当前文创产品普遍采用预先设计制作模式，难以满足游客个性化需求的供需矛盾，探索相关技术在中文文创领域的应用潜力。此外，本项目还致力于为旅行者等用户提供便捷的工具，使其能够随时随地进行个性化文创设计和制作，满足市场需求。

2.1.2 相关技术现状

传统的图像编辑软件 (如 Adobe Photoshop, Illustrator): 这些软件功能强大, 可以实现对图片的文字进行修改和添加。但它们通常需要专业技能, 且操作相对复杂, 难以满足普通用户快速、便捷地进行个性化文创设计的需求。此外, 这些软件在生成与背景自然融合的文字方面也存在一定的局限性。

在线设计平台 (如 Canva, 稿定设计): 这些平台提供了丰富的模板和素材, 用户可以进行简单的文字替换和排版。但其个性化定制程度相对较低, 难以实现高度自由的创意表达。

已有的文字控制图像生成模型 (如 GlyphDraw, Textdiffuse, AnyText): 这些模型在解决字体与背景融合方面取得了一定的进展, 但正如前文所述, 它们仍然难以完全避免文字生成中的错误, 并且缺乏专门针对中文的优化。当前模型通过集成大语言模型提升了文本生成的稳定性, 然而, 对文本生成位置的精细化控制以及基于图像内容的文本引导修改能力仍有待提升。

2.1.3 项目的主要开发工作 (具体参见系统设计概述文档)

为解决上述问题, 本项目的主要开发工作包括:

- **模型微调与优化:** 采用 AnyText 模型作为基础, 并针对中文应用场景进行专门的训练和优化。具体分为文字控制框架的训练和扩散模型的训练两部分。
- **数据集构建:** 制作了两份数据集。一份是基于 AnyWord-3M 数据集筛选出的约 40 万条数据, 用于微调 AnyText 模型。另一份是通过网络爬虫采集的约 1000 张与中华文化及文物相关的图片, 用于微调 stable diffusion v1-5 模型。
- **系统实现与部署:** 基于 Gradio 搭建了交互式的网页界面, 实现了文字到图片生成和图片文字编辑两大核心功能。

2.1.4 项目成员、分工及完成情况

成员	主要分工	完成情况
郑仕博	• 项目统筹与规划 • 模型训练与调优 • 核心代码编写与上传 • 项目部署	• 负责修订项目计划书并明确团队分工 • 专注模型训练, 研究并解决过拟合问题 • 与陈奕嘉共同进行扩散模型的训练 • 完成最终版模型的训练, 并上传至 ModelScope • 编写模型测评文件代码并上传至 GitHub • 与组员共同完成前后端制作并上传至 GitHub • 使用 LaTeX 重新书写了系统概述和需求分析文档 • 参与编写 Dockerfile, 将项目封装并上传至 Docker Hub • 将项目部署至 ModelScope • 完善文档, 增加技术难点部分, 并制作测试报告和 PPT

陈奕嘉	• 文档撰写与管理 (项目计划书、需求分析、软件著作权等) • 数据集搜集与整理 • 辅助模型训练与调节 • 软件测试	• 起草并完成项目计划书和需求分析书初稿 • 负责搜集与中华优秀传统文化主题相关的训练集 • 合并了两个模型的权重 • 与郑仕博一同完成模型部分的训练 • 撰写软件著作权说明书与项目注意事项的初稿 • 使用 LaTeX 修改系统设计文档 • 参与编写 Dockerfile 并封装项目 • 参与将项目部署至 ModelScope • 筹划并执行软件测试 • 参与制作最终汇报 PPT

苏泳豪	• 前后端框架开发与实现 • 数据集搜集与标注 • 辅助模型调试 • 项目部署支持	• 寻找并修改了前后端模板, 确定基本框架 • 负责收集并整理所有成员找到的数据集 • 编写脚本对标签进行修改, 并标注数据集 • 完成前后端代码的书写, 实现了文本生成和图像编辑两大核心功能 • 使用 LaTeX 重新排版需求分析文档 • 参与编写 Dockerfile, 解决网络问题, 成功将项目封装并上传至 Docker Hub • 编写了 requirements.txt 文件, 并参与将项目部署至 ModelScope • 对项目产品进行测试, 并参与制作汇报 PPT 和完善文档

表 1: 项目成员、分工及完成情况

2.2 项目策划、分析与设计工作情况

2.2.1 项目立项计划及完成情况

项目旨在解决文创产品个性化不足以及现有技术在中文处理上的缺陷。项目计划通过微调先进的生成式 AI 模型, 开发一个能够让普通用户轻松设计个性化文创图像的工具。目前, 项目已经完成了模型的训练、功能实现、系统部署, 并申请了软件著作权。

2.2.2 系统需求分析工作和结果

功能需求 系统主要提供两种创作方式:

- **基于用户输入的文字:** 用户输入文字内容, 在预览区指定文字位置和大小, 生成创意图片。

- **基于用户输入的成形图片：**用户上传图片, 在图上标注文字位置并输入内容, 系统生成最终的文创图片。

两种方式都支持结果的预览、调整、保存和分享, 具体可见使用手册。

性能需求　模型需要具备较高的文本准确性、卓越的图像生成能力, 并且能够避免过拟合, 保证良好的泛化性能。此外, 该系统对硬件有一定的要求, 如下:

表 2: 硬件要求

类别	**基本要求**
服务器端	Intel Core i5(或更新)； 内存 32G 以上； GPU NVIDIA RTX 4060, 内存 8G 以上； 硬盘剩余空间不低于 50G；
客户端	能运行现代网页浏览器即可

2.2.3　系统设计工作及结果

本系统基于阿里云开源的 AnyText 模型框架进行开发与扩展, 整体架构由三个核心模块组成: 辅助潜在模块 (Auxiliary Latent Module)、文本嵌入模块 (Text Embedding Module) 以及文本控制扩散生成模块 (Text-control Diffusion Pipeline)。在此基础上, 我们对文字控制能力和图文融合性能进行定向增强, 并完成了多项训练与优化工作。

技术框架　本系统使用 AnyText 模型, 结合了 SD1.5 的图像扩散模型与文本控制机制, 能够实现图像中文字的精确生成与编辑。模型架构如下 (图1):

- **Auxiliary Latent Module**: 处理输入的文字渲染特征, 与图像潜变量融合, 指导扩散过程。
- **Text Embedding Module**: 将文字内容编码为向量特征, 结合 OCR 与 Tokenizer 保持语义一致性。
- **Text-control Diffusion Pipeline**: 基于 UNet 和控制网络实现文字引导的图像生成, 并引入 Text Perceptual Loss 实现图文融合感知优化。

该架构支持文字生成图与图像中指定区域的文字编辑, 包括添加、修改与删除, 具有良好的图文融合能力。

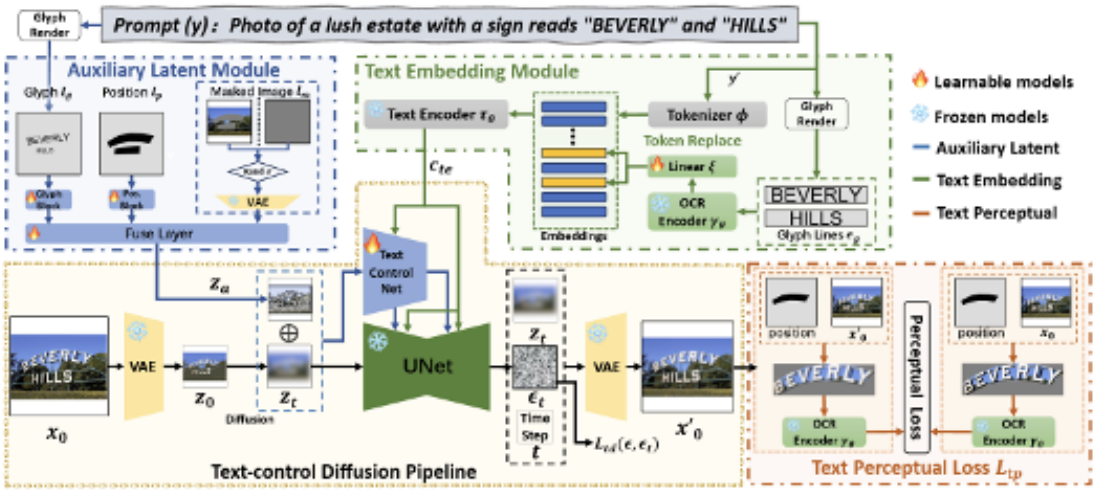

图 1: 模型示意图

界面设计　提供 Web 端界面, 两种主要操作模式:

1. 文到图像的生成: 输入提示词, 并将需要渲染的文本用 “” 标注, 可以通过画布绘制文本位置、拖框选择文本位置或随机选择文本位置;

2. 图片文字编辑, 手动掩盖需要修改区域, 输入文本并进行编辑。

界面上有说明、参数设置、文本输入框、模式选择、文字位置标注、样例 (点击即可)、运行按钮、图片结果展示和加强训练的物品, 用户可调整 CFG-Scale、Steps 等参数, 查看结果并保存。详情见图2和图3。

详情见系统设计概述文档, 用户使用手册。

2.2.4　系统具体实现工作及结果

数据集制作　本项目制作了两份数据集, 第一份数据集用于微调 AnyText 模型, 另一份数据集用于微调 stable diffusion v1-5。

- **数据集来源:** 本项目利用爬虫技术从 Google、Bing、百度等搜索引擎抓取数据, 自动化访问网站并提取网页内容 (如文本、图片、链接等), 筛选并保存有价值的信息。项目重点聚焦于文物图像数据的采集, 旨在抓取并整合图文结合的信息, 以构建高质量的数据集。大规模多语言数据集 AnyWord-3M, 于论文 AnyText 中提出。数据来源涵盖 Noah-Wukong、LAION-400M 及多个 OCR 任务数据集 (如 ArT、COCO-Text、RCTW 等), 覆盖街景、书籍封面、广告等多种文本场景。OCR 数据直接利用已有标注, 其余图片经 PP-OCR 处理并由 BLIP-2 生成文本描述。经过严格筛选与后处理, 最终获得 303 万余张图片, 包含 900 万行文本及 2000 万余字符。

- **数据规模**第一份基于 AnyWord-3M, 按照水印、文字块、语言 (保留大部分中文数据集, 少部分英文数据集) 等标准精筛后, 保留约 40 万条数据。第二份数

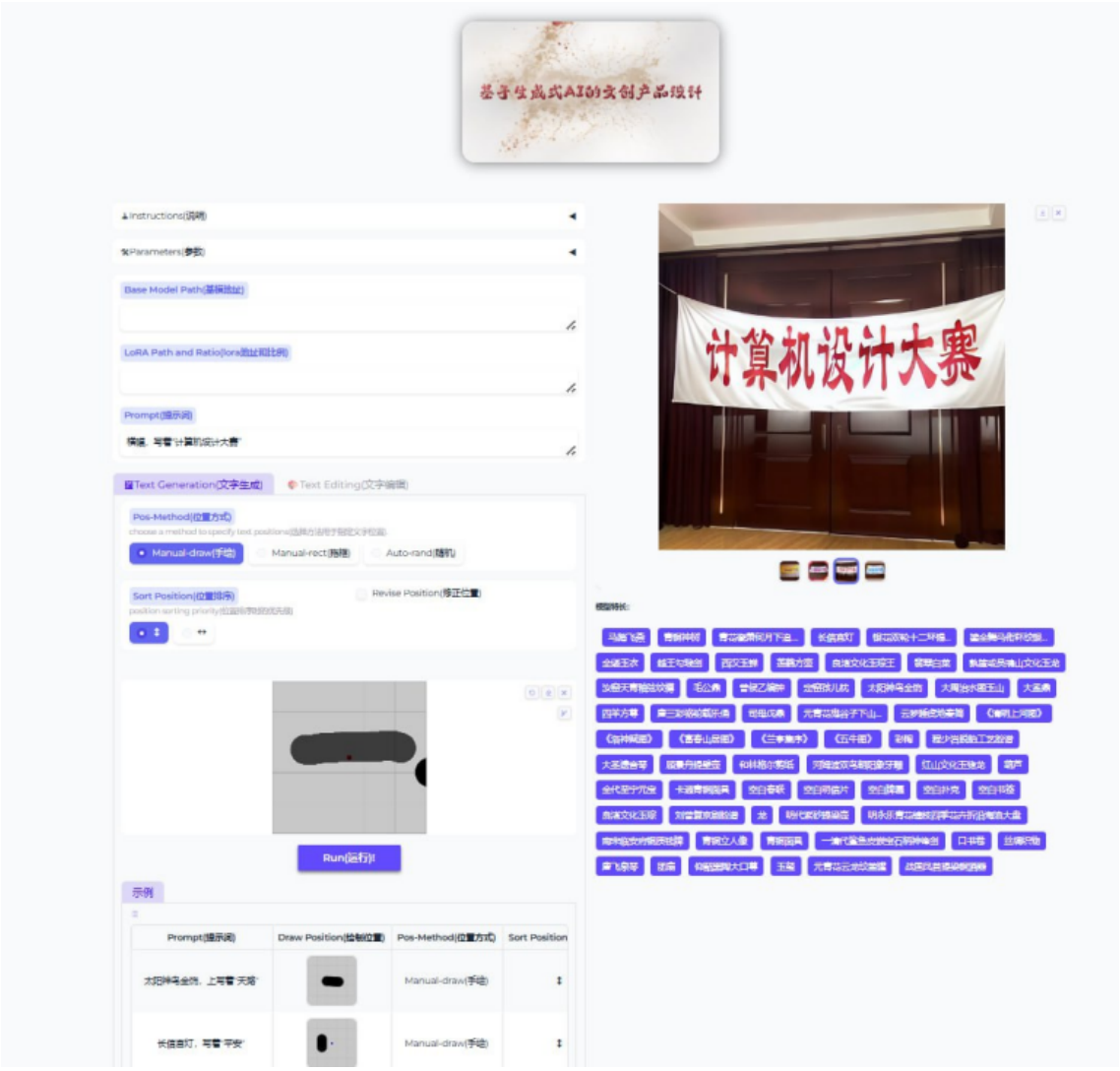
基于生成式AI的文创产品设计
Instructions(说明)
Parameters(参数)
Base Model Path(基模地址)
LoRA Path and Ratio(lora地址和比例)
Prompt(提示词)
Text Generation(文字生成)
Text Editing(文字编辑)
Pos-Method(位置方式)
Manual-draw(手绘)
Manual-rect(拖拽)
Auto-rand(随机)
Sort Position(位置排序)
Revise Position(修正位置)
Run(运行)!
示例
Prompt(提示词)
Draw Position(绘制位置)
Pos-Method(位置方式)
Sort Position
Manual-draw(手绘)
Manual-draw(手绘)
计算机设计大赛

图 2: 人机界面 1

图 3: 人机界面 2

据集由爬虫采集约 1000 张与中华文化及文物相关的图片, 并通过水印筛选和修整优化质量。随后, 使用 wd14-convnextv2-v2 进行自动标注, 并对标注结果进行了适当调整, 以提升准确性和适用性。

模型微调 针对中文场景, 本项目将对 AnyText 模型先与 Realistic_Vision_V4.0 (基于 sd1.5) 进行权重的合并, 在中文训练集上的专门训练, 使模型更好地适应中文应用场景, 确保其在实际应用中展现卓越的表现能力。此外为了更贴合主题, 本项目在 Realistic_Vision_V4.0 (基于 sd1.5) 权重基础上使用 dreambooth 的方法进行微调, 并与微调后的 AnyText 模型的权重进行整合。(第一步的 sd1.5 只是用于训练的, 其权重并不会改变, 而第二步才微调 sd1.5 然后进行权重的整合) 第一步使用第一份数据集, 第二部采取第二份数据集。

部署与框架实现 本项目将使用 Gradio 搭建模型的交互式部署框架。结合任务需求, 设计并开发直观易用的网页和交互界面, 便于用户体验模型的功能和效果, 同时支持进一步的迭代优化。

2.3 系统部署与测试工作情况

2.3.1 系统部署与调试

系统通过 Gradio 成功部署, 提供了一个用户友好的 web 界面, 支持文字生成图片和图片编辑功能 (可以使用 Docker)。

2.3.2 系统测试

系统测试使用的是 AnyText Benchmark, 包含文字生成和修改两部分。测试结果表明, 模型在文字生成和编辑方面表现良好, 但在文字控制框架的训练过程中, 由于算力限制, 训练效果未达到预期。

- **文字正确率测试:** 采用 OCR 识别与标注对比的方式进行评估。训练后的模型在文字生成和修改的正确率及编辑距离上没有显著提升, 分析认为与算力、参数及训练批次有关。因此, 项目最终采用了官方的 AnyText 权重。

表 3: 文字控制框架评估结果

状态	正确率 ↑	编辑距离 ↓
文字生成 (训练前)	0.6957	0.8402
文字生成 (训练后)	0.6644	0.8282
文字修改 (训练前)	0.6671	0.8298
文字修改 (训练后)	0.6644	0.8282

表 4: 扩散模型评估结果

状态	FID ↓
训练前	31.558
训练后	34.242

- **扩散模型评估:** 采用 FID (Fréchet Inception Distance) 进行评估。训练前后的 FID 值接近, 证明模型没有发生严重的过拟合。
- **结论:** 因此目前项目使用官方 AnyText 的权重并与训练好的扩散模型进行合并, 得到最终的模型权重。

2.3.3 生成对比测试

见图4和图5。测试结果表明, 模型在生成与背景自然融合的文字方面表现良好, 能够满足用户个性化文创设计的需求。

图 4: 生成对比测试 1

2.4 项目管理相关工具使用情况

本项目在管理和协作过程中, 充分利用了多种项目管理与协作工具, 提升了团队的沟通效率和项目推进效率。具体如下:

- **代码与文档管理:** 项目核心代码、模型测评文件、前后端代码、系统文档等均托管于 **GitHub**, 实现了版本控制和团队协作开发。

图 5: 生成对比测试 2

- **模型与容器管理:** 训练完成的模型上传至 **ModelScope**, 便于模型的共享与部署；项目容器化后上传至 **Docker Hub**, 实现跨平台部署和环境一致性。
- **文档与报告编写:** 项目计划书、需求分析、系统设计等文档采用 **LaTeX** 进行协作撰写, 保证了文档的规范性和可维护性。
- **任务分工与进度管理:** 团队通过定期会议和在线协作工具明确分工, 及时同步进展, 确保各项任务有序推进。
- **测试与评估:** 项目测试报告、PPT 等成果文档也通过 GitHub 进行版本管理和共享。

经费使用情况表格显示, 项目主要开销用于租用云服务器和 GPU 资源, 共计 1513.22 元。

2.5 项目讨论与体会

2.5.1 对项目过程的体会

- **作品特色与创新点:** 项目实现了高度的个性化定制, 操作便捷, 生成的文字能与背景自然融合。其创新点在于通过创新的文字渲染模型, 革新了图片修改的应用场景, 并支持对图片中文字的便捷编辑。
- **挑战与不足:** 在文字控制框架的训练过程中, 由于算力限制 (至少需要 8 卡 V100), 训练效果未达到预期, 这是一个主要的挑战。此外, 该模型缺少对文字更多方面的控制, 如字体、颜色等, 且对中文的支持仍有待加强。

2.5.2 项目和项目管理过程的优点与不足

优点

1. **高度个性化与创新性:** 项目的核心创新点在于实现了高度的个性化定制, 能够让用户轻松设计独特的文创图像作品。通过创新的文字渲染模型, 它革新了图片修改的应用场景, 并支持对图片中文字的便捷编辑, 满足了当前市场对个性化文创产品的迫切需求。

2. **技术路线清晰且先进:** 项目基于阿里云开源的 AnyText 模型框架进行开发与扩展, 结合了 SD1.5 的图像扩散模型与文本控制机制, 技术选型先进且合理。通过模型微调与优化, 使其更好地适应中文应用场景, 展现了我们在 AI 前沿技术应用方面的能力。

3. **系统功能完善:** 系统实现了文字到图片生成和图片文字编辑两大核心功能, 并且通过 Gradio 搭建了直观易用的 Web 交互界面, 方便用户体验。同时, 项目还支持结果的预览、调整、保存和分享, 功能设计全面。

4. **成果可转化性强:** 项目已产出实际可用的系统, 并申请了软件著作权。这表明项目不仅停留在理论研究层面, 更具备实际应用和市场推广的潜力。

5. **规范化管理与协作:** 项目在管理和协作过程中充分利用了多种项目管理与协作工具, 如 GitHub 进行代码与文档管理,ModelScope 和 Docker Hub 进行模型与容器管理,LaTeX 进行文档编写, 提升了团队的沟通效率和项目推进效率。

不足

1. **计算资源受限:** 项目在关键的模型训练环节受到了计算资源的严重制约, 特别是文字控制框架的训练, 由于算力限制 (至少需要 8 卡 V100), 训练效果未达到预期。这导致了文字生成正确率没有显著提升。

2. **模型功能仍有提升空间:** 现有模型在文字更多方面的控制上仍有待加强, 例如缺乏对字体、颜色等更精细的控制。同时, 对中文的支持仍需进一步优化。

3. **部署优化空间:** ModelScope 上无法持久化存储的问题导致每次启动速度较慢, 影响了用户体验, 未来需要进一步解决部署的效率问题。

展望 尽管存在一些不足, 但本项目为未来的发展奠定了坚实的基础。我们已计划未来采用性能更优的 AnyText2 模型, 该模型增加了字体选择功能, 有望解决当前模型对字体控制不足的问题。同时, 我们计划在编辑板块补充更多精美的图片素材, 以激发用户创作灵感, 进一步提升用户体验。这些展望表明我们对项目的未来发展充满信心, 并已明确了后续的优化方向。

2.6 参考资料

- 软件项目计划书 (submit)
- 需求分析
- 系统设计概述
- 基于生成式 AI 的个性化文创图像作品设计系统说明书
- 个人周报 (内含问题报告)
- 报告 PPT