附件 2

序号：_____

编码：_____

武汉大学第十六届“自强杯”
大学生课外学术科技作品竞赛

作品正文及附件材料

作品名称：笔韵智枢—大数据驱动的智能一体化字体创作引擎

学院名称：国家网络安全学院

申报者姓名：

（集体名称）：“笔韵智枢”团队

类别

☐自然科学类学术论文

☐哲学社会科学类社会调查报告

☐科技发明制作 A 类

☑科技发明制作 B 类

武汉大学第十六届“自强杯”大学生课外学术科技作品竞赛组委会制

目录

第 1 章 作品概述 **1**

第 2 章 问题分析 **2**
2.1 问题来源 . . . 2
2.1.1 政策与背景 . . . 2
2.1.2 现状与困境 . . . 3
2.2 现有解决方案 . . . 4
2.2.1 字体制作——传统方法 . . . 4
2.2.2 字体制作——批量制作 . . . 5
2.3 本作品要解决的痛点问题 . . . 6
2.3.1 字体设计——痛点总结 . . . 6
2.3.2 基于 AI 生成的方法——三大挑战 . . . 6
2.4 解决问题的思路 . . . 8

第 3 章 技术方案 **10**
3.1 模型总览 . . . 10
3.2 核心生成模块——基于多头编码器 . . . 10
3.2.1 多头编码器 . . . 11
3.2.2 风格与组件特征分类器 . . . 12
3.2.3 生成器 G . . . 13
3.3 参考字数据优化模块——基于四角编码映射 . . . 14
3.3.1 参考字列表的动态更新 . . . 14
3.3.2 基于四角编码的最大化距离向量算法 . . . 14
3.4 源字体数据优化模块——基于 ResNet-18 网络 . . . 17
3.4.1 源字体匹配模块原理与构建 . . . 17
3.4.2 风格匹配器部署与应用 . . . 18
3.5 模型训练数据——古今结合助力模型优化 . . . 20
3.5.1 思路概览 . . . 20
3.5.2 现代字体库 . . . 20
3.5.3 生僻字库 . . . 21
3.5.4 古书法字库 . . . 22
3.6 专业外设支持——一站式解决方案 . . . 23
3.6.1 硬件系统总体架构 . . . 23
3.6.2 绘图输入——手写笔和触摸屏 . . . 23
3.6.3 字体生成——GPU 设备 . . . 24

第 4 章 系统实现 **25**
4.1 技术栈总览 . . . 25
4.2 软件上层设计 . . . 25
4.2.1 程序主体分析 . . . 25
4.2.2 用户界面设计 . . . 26
4.3 模块部署 . . . 31

4.3.1 基于多头编码器的字体生成模块 . . . 31
4.3.2 基于四角编码映射的参考字数据优化模块 . . . 33
4.3.3 基于 ResNet-18 网络的源字体数据优化模块 . . . 34
4.4 硬件接口设计 . . . 35

第 5 章 测试分析 37
5.1 测试计划描述 . . . 37
5.2 功能测试 . . . 37
5.2.1 基本功能测试 . . . 37
5.2.2 可靠性测试 . . . 38
5.2.3 易用性测试 . . . 38
5.3 性能测试 . . . 39
5.3.1 评价指标介绍 . . . 39
5.3.2 效果测试一：核心生成模型生成效果测试 . . . 39
5.3.3 效果测试二：数据优化模块对生成效果影响的消融实验 . . . 40
5.3.4 效果测试三：大数据对模型生成效果影响的消融实验 . . . 41
5.3.5 成本测试一：AI 生成与传统方法消耗成本对比测试 . . . 42
5.3.6 成本测试二：参考字数据优化模块对成本开销影响的消融实验 . . . 42
5.3.7 成本测试三：源字体数据优化模块对成本开销影响的消融实验 . . . 42
5.3.8 成本测试四：工具在不同硬件设备上的运行成本测试 . . . 43

第 6 章 作品总结 44
6.1 作品特色与创新点 . . . 44
6.1.1 三大特色 . . . 44
6.1.2 三大创新 . . . 45
6.2 应用推广 . . . 45
6.2.1 实地部署——赋能小企业小工作室字体创作 . . . 45
6.2.2 落地合作——腾讯输入法“汉字守护计划” . . . 46
6.3 作品展望 . . . 47
6.3.1 应用前景 . . . 47
6.3.2 后续升级方向 . . . 47

参考文献 48

第 1 章 作品概述

汉字是中华民族的文化瑰宝。进入了信息化时代，字库产业成为为汉字文化在数字世界中的载体。各产业的发展和国家政策都对字库产业提出了高标准和严要求。但是目前字库产业还面临着字体制作难的产业痛点。目前逐字设计的字体创作方法流程繁琐，耗费成本高，而基于开源字体修改的方法则面临着风格同质化等局限。

针对现有行业痛点，本团队独创性地采取 AI 批量生成字体的思路，用户只需要提供 10 个参考字，便可通过 AI 模型自动生成带有参考字风格特征的目标字体。但是 AI 字体生成方法也面临着参考字结构多样性、风格杂糅，以及复杂字形上的泛化能力三大挑战。本团队提出基于大数据的模型优化方法，在该任务上做出了以下三大创新：

- **一个核心生成模型：**设计了专用于字体部件特征抽取的多头编码器，并使用匈牙利算法来匹配汉字组件。将源字体的结构特征和参考字的风格特征用于目标字体的生成。
- **两大数据优化算法：**设计了基于四角编码映射的参考字推荐优化算法和基于 ResNet-18 网络的源字体匹配优化算法，并且使用三大类型数据集进行模型构建。从数据角度同时优化了用于体验与生成效果。
- **三大类型数据：**构建了现代字体库 + 生僻字库 + 古书法字库三大类型的数据集，包含 12 大类 500 余种不同风格的字体。用以数据优化模型的构建和核心生成模型的训练。实现“大数据助力小样本生成”的构想。

本团队将所提出算法落地，设计了简洁易用的软件界面，并适配专业外设以满足专业设计的需求。作品已经进入实地部署应用阶段，应用结果表明“笔韵智枢”工具具有功能专业、使用便捷、生成精准的特点。将原先长达数月的字体设计周期，缩短到十几分钟以内；而相比起传统的批量生成方法，本工具的生成效果具有高达 30% 的提升。大大降低了字体制作的成本和门槛，使得字体制作不再困难。

第 2 章 问题分析

2.1 问题来源

2.1.1 政策与背景

自上古仓颉造字的传说以来，汉字伴随着中华文明走过了千百年的漫长岁月。汉字文化是中华传统文化亮丽的剪影。时间来到信息化时代，包括了字体设计、字体授权、字体应用在内的字库产业成为了汉字文化在数字世界中的载体。自 2019 年以来，我国字库产业市场规模呈逐年上升趋势。

图 2.1: 2019-2023 年中国字库产业市场规模趋势

从广告设计到艺术影视，字库在产品的视觉传达方面起到了举足轻重的影响。一款具有其独特风格的字体能够赋予产品鲜明的标志。与此同时，丰富而多样的字体风格也是中国博大精深的书法文化的重要组成部分。

图 2.2: 字库产业在各产业中的应用

新时代文化建设被纳入“五位一体”总体布局 [1]，字库产业建设作为文化建设中的重要一环，在其中扮演着不可或缺的角色。其中新的汉字字符集强制性国家标准 GB18030-2022[11] 已在 2022 年 8 月起实施，对语言文字的规范化、标准化、信息化提出了更高的要求。

时间	政策名称	关键词
2015年	《关于推广应用国家通用字形表的通知》	推广通用字形表 规范字体设计
2019年	《关于加强版权保护的意见》	字体知识产权保护
2020年	《国家通用语言文字普及提升工程和推普助力乡村振兴计划实施方案》的通知	政务数字化建设 字体应用广泛性
2022年	汉字字符集强制性国家标准GB18030-2022	规范化、标准化、信息化

图 2.3: 2015 年来有关汉字字库的国家政策

2.1.2 现状与困境

虽然字库产业在文化传承、经济发展方面都担任着重要的责任，现有政策也对该产业的发展提出了激励与指示，但是经过我们团队的走访调研，发现目前的字库产业仍面临着字体库匮乏、垄断严重、版权问题多、生僻字难支持、古代书法难保护等困境：

图 2.4: 我国字库产业目前面临的五大困境

- **字体库匮乏：**目前汉字数字化字体仍相比于其他语言字体较匮乏，仅占全世界字体库的 2%
- **垄断严重：**目前中国字库产业份额主要集中在汉仪、方正、华文等几家头部企业，其拥有着内容、技术、人才、资金等方面的优势，现存中文字体中 90% 的版权属于头部企业。
- **版权问题多：**对于一些个人或是企业，在无法自己创作字体时，往往会选择从网上下载字体使用，一旦误用了未授权商用的字体作为商用，实际上已经涉嫌字体侵权。
- **生僻字难支持：**目前有史料可查的汉字已经超过了 30 万字，但是我们的常用字符集大多还停留在不到 1 万字的层面。
- **古代书法难保护：**进入信息时代，如何将存在于古籍、古碑上的书法文字修复并存档为数字遗产，是书法文物保护所面临新的挑战。

究其根本，以上这些困境的出现，归根到底是因为中国汉字数量大，**数字化字体的制作难**，字体创作是字库产业的生命力和活力之源，只有解决字体创作难的问题，字库行业才能涌入更多的新鲜血液。

2.2 现有解决方案

本团队分别到方正、汉仪等头部字库企业和一些小型设计工作室进行走访调查，发现目前主流的字体创作方法分为传统逐字设计的方法和较新的批量制作两类，以下将对这两种方法各自的特点与局限进行详细介绍。

2.2.1 字体制作——传统方法

字体创作的传统方法指的是对字体中的每个汉字，逐字设计其风格和外观，该方法是目前原创字体最常采用的一种方法，其优点在于字体呈现效果精致可控，个性化程度高，但是其最主要的局限在于需要对字库中的每个汉字做手工设计和调整，一款字体经过逐字绘制方法创作的流程如下：

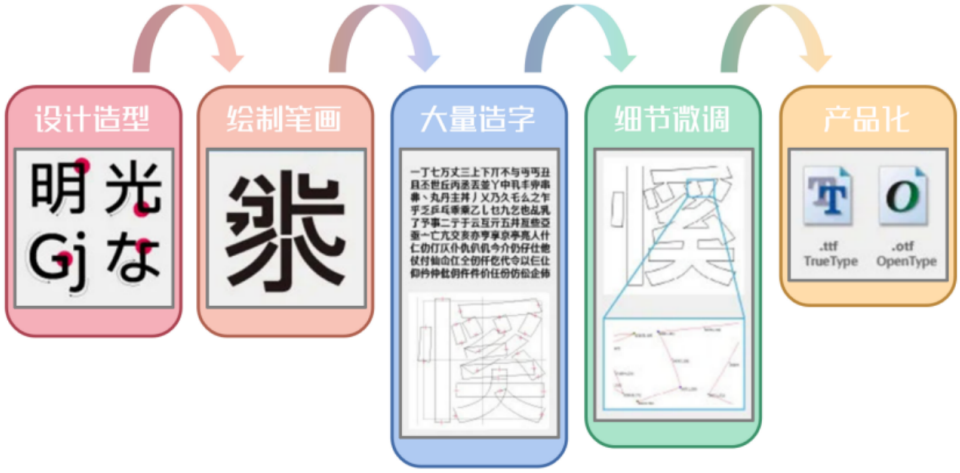


图 2.5: 逐字设计的字体创作方法的五大流程

在大量造字阶段，设计师需要在专业字体设计软件之中将提前设计好的笔画逐一按照字形结构进行排列堆砌。而在细节微调部分设计师需要利用锚点工具修饰字形上的细节，使得字体的规格统一。

经过我们的调查，使用传统的逐字设计方法制作一款成熟的字体所需要投入的人力、技术、资金成本巨大：

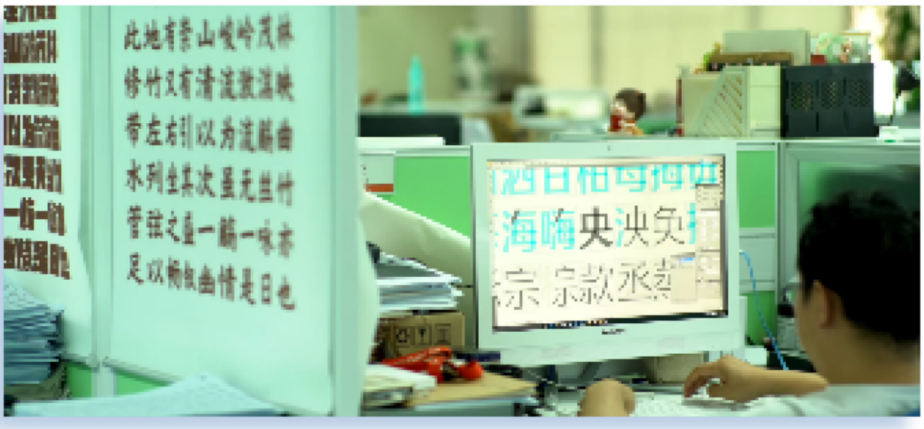


图 2.6: 专业设计师逐字设计字体

综上，目前使用传统方法逐字设计字体，**制作周期长、人力需求大、技术门槛高、资金成本大**，头部大企业尚且需要投入巨大的成本以及经过繁琐的流程，而对于个人或是小工作室，面对如此高的成本，也只能在字体设计领域的大门前望而却步。

调研项目	调研结果
制作周期	少则一年，多则数年
制作团队	5名以上的专业设计师
技术要求	熟练掌握photoshop、illustrator、FontLab等专业设计工具
投入成本	数十万甚至百万元

图 2.7: 逐字设计一款字体所需要投入的各方面成本

2.2.2 字体制作——批量制作

批量制作方案是近年来字体制作领域一类更新更高效的解决方案，该方案可以概括为在已有的开源/可商用字体的基础上，进行风格上的微调，批量生成最终的字体。这类方法的优点在于批量生成，相比于传统方法的时间人力成本小。效率更高。

开源字体指的是源码（描述了字体的轮廓、曲线形状等特征）公开的字体，其允许任何人在遵循其许可证的前提下修改和定制该字体，在此基础上进行二次开发和创作。开源字体通常以字体文件的形式发布，其中包含了字形的矢量数据。设计者可以使用 FontForge 等工具来对开源字体进行修改，包括调整其粗细、斜度、曲率、轮廓等特征，如下图所示。

图 2.8: 在开源字体的基础上进行二次创作

虽然基于开源字体的字体设计大大简化了字体创作的难度，但是因为从头设计一款新的开源字体需要耗费更大的工作量，所以该方法面临着开源字体数目少、风格种类少的局限。截止 2023 年 4 月，中文开源字体集 [12]（OSFCC）之中仅收录了 100 余种基础开源字体，而目前市面上的开源字体大多数也是由这些基础开源字体调整改变而来。与此同时，我们调查统计了这些基础开源字体中各种类型字体占比，如下图所示：

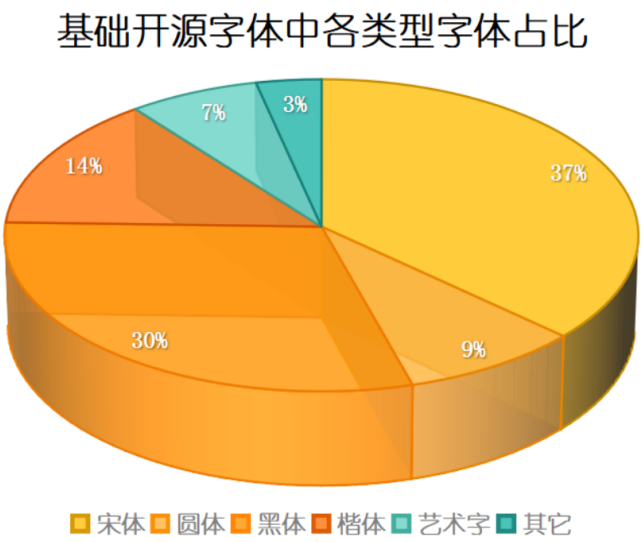


图 2.9: 对中文开源字体集中的开源字体仍以基础风格为主，同质化严重

可以看到，目前的基础开源字体中大部分是宋体、黑体、楷体等基本类型，而对于创作空间和艺术设计需求更大的艺术字、书法体、手写体，这些类型的开源字体仍面临着较大的空缺。

由此可见，因为目前中文开源字体仍然面临着**字体数目少**、**字体风格同质化**等局限，并且开源字体设计工具的**技术门槛也较高**。这导致当开发者想基于开源字体来实现自己的艺术创意时，面临无对应基础风格的开源字体可用的窘境。

2.3 本作品要解决的痛点问题

2.3.1 字体设计——痛点总结

字体设计在字库产业中扮演着整个产业的生命力源泉的重要角色，但是以上提到的几种字体设计解决方案，仍分别存在着许多局限，导致了字体创作难的产业现状。我们从目前主流的解决方案触发，总结了字体设计领域现存的痛点如下：

- **逐字设计——成本高、周期长：**因为汉字规模的巨大，逐字设计的传统方法在大量造字和细节微调部分需要经历极其繁琐的设计工作，并且需要专业设计师使用专业软件来进行制作，该方法面临着**制作周期长**、**人力需求大**、**技术门槛高**、**资金成本大**的严重痛点。
- **批量制作——局限大、效果差：**基于开源字体的批量制作方法是近年来兴起的另一种主流思路，虽然批量生成大大简化了逐字设计的繁琐流程，但是因为目前**开源字体少，且风格同质化严重**，设计师在有创作灵感时，使用该方法往往不能将其完美地实现，另外开源字体制作修改工具使用的技术门槛也较高，给创作者带来了巨大的学习成本。

综上，字体设计工作者急需一种同时满足**字体制作快捷**、**工具上手简单**、**设计效果精良的**解决方案，这也是字体制作难这一产业根本问题的解决之道。

2.3.2 基于 AI 生成的方法——三大挑战

AI 生成是解决字体设计领域现存痛点的一种新的思路，生成式对抗网络已经在各种图像生成任务中表现出了优秀的性能，包括图像风格迁移。基于参考字-源字体的图像风格迁移任务，可以用于批量字体创作工作中，**不仅能够省去逐字设计的繁琐流程，还突破了批量生成现有的风格限制**。其所需数据和功能的简要介绍如下表所示：

数据	细节
参考字（reference font）	作为所需要生成的目标字体的示例， 其规模较小，但是带有所要求生成的风格特征
源字体（source font）	提供完整汉字集合中每个汉字的结构信息， 其规模较大，模型本质上将参考字的风格迁移到该字体上，以生成对应的目标字体
目标字体（result font）	AI 字体生成成模型的输出， 带有参考字风格特征的完整字汉字集合

表 2.1: 字体相关数据说明

借助基于 GAN[13] 的风格迁移网络，可以在仅提供少量带有目标风格的参考字的前提下，将该风格特征迁移到另一种字体的完整汉字集中，实现个性化的批量字体生成。但是该思路目前仍不成熟，在具体部署和实现中仍有许多挑战，经过充分的实验和尝试，我们将该方法面临的技术挑战总结为以下三点：

- **参考字型多样性的挑战：**因为参考字实际上代表了所需要生成的全部字体的风格，所以我们需要让参考字尽可能地携带更多偏旁部首，以确保最终的生成效果，如下图所示，如果提供的参考字清一色都是类似的字形，则会使得模型在其他字形上的生成效果不佳。在实验过程中，我们发现对于目前的风格迁移网络，至少需要按要求提供 200~300 个参考字，才能达到较为优良的生成效果，**如何尽可能地实现少量参考字提供完整字形部件信息，是目前 AI 生成字体面临的一大挑战**。

图 2.10: 带有更多样字体部件的参考字集合会携带更丰富的风格参考信息

- **风格杂糅问题：**模型将参考字的风格特征迁移到源字体的结构上，当参考字与源字体的风格差异较大时，所生成的目标字体便会同时带有两种字形的风格，该现象被称为风格杂糅，如下图所示，当发生风格杂糅时，所生成的字体实际上是不可用的。在实际实验之中，往往需要人工进行多次尝试，才能将字体生成风格杂糅的现象最小化，**如何在算法层面抽取并识别参考字的风格，并迅速为其匹配出风格最相似的源字体，是目前 AI 生成字体面临的第二大挑战**。

图 2.11: 产生风格杂糅时的生成效果远远逊色于正常情况

- **复杂字体/非典型字体上的泛化能力：**目前基于传统 GAN 的风格迁移模型 [14] 仅在笔画简单的字形上表现较好，而具有复杂结构/生僻字或是古文字上，如下图所示，模型的泛化能力仍不理想。**如何构建模型，使生成目标字体时能更精确地地专注于汉字各部件的风格，并且构建带有更多样化字体字形的训练数据集，是目前 AI 生成字体面临的第三大挑战**。

图 2.12: 传统 GAN 生成字体：复杂字形下瑕疵较多

2.4 解决问题的思路

针对字体设计行业目前存在的传统方法流程繁琐成本高和批量生成风格局限效果差的痛点，本团队采用了 AI 生成字体的新思路，用户仅需要提供少量参考字，便能生成带有其对应风格的完整字体。

面对 AI 生成字体模型面临的字型多样性、风格杂糅、泛化能力差的三重技术挑战，本团队**借助大数据思想助力模型构建**，独创性地提出了“**1 个核心模型 +2 个数据优化模块 +3 大类型数据集**”一站式创作工具的解决方案：

- **1 个核心生成模块——基于多头编码器：**在传统 GAN 网络的基础上，我们构建了基于多头编码器的少样本字体生成模型。模型使用**多头编码器**和**匈牙利算法**来实现局部特征提取与特征自动匹配，使模型能更好地适配更大和更多样化的数据集。实验证明相比于传统 GAN 网络，本生成模型在字体平均相似度（LPIPS）和真人辨别率上分别有 33.94% 和 26.79% 的提升，在生僻字域上，提升高达 56.30% 和 67.17%。**使得 GAN 网络所生成的字体能真正意义上的投入应用**。
- **2 个数据优化模块——参考字 + 源字体：**针对参考字组件多样性和字体风格杂糅问题，我们分别设计了基于**四角编码映射**的参考字推荐优化算法和基于 **ResNet-18** 网络的源字体匹配优化算法，并且使用三大类型数据集进行模型构建。消融实验证明，这两个数据优化算法对模型的 LIPIS 分别带来了 5.3% 和 6.9% 的提升，而在软件应用角度，使用两个数据优化模块，使一款字体的制作周期平均缩短了 90% 以上。**从数据角度同时优化了用户使用体验和模型生成效果**。
- **3 大类型数据集——古今结合：**针对模型在复杂、非典型字形上生成效果不佳的技术挑战，我们爬取了 12 大类 500 余种字体的**现代字体库**、收集并标注了 5 种字体 10000 余字的**生僻字库**和 69 部古代书法著作共 5000 字的古书法字库，三大类型数据共同构建起实现模型优化的数据集。实验证明相比起传统的汉字字体库数据集，该方法使得最终模型生成字体的 LPIPS 获得了 27.54% 的提升。使用**大数据**优化**小样本**生成，是本团队所提出的最大创新。

基于 pyGame 框架统一上述三个模块，本团队设计了集合参考字绘制、参考字推荐、源字体匹配、少样本字体生成为一体的一站式字体创作工具，软件使用上手简单，并且适配了数位板、手绘笔等专业外设，满足了专业设计的需求。

第 3 章 技术方案

3.1 模型总览

本作品面向字体设计师，项目核心包括以下三个模块：

- 使用 3 大类型数据集训练的字体生成模块
- 基于四角编码映射的参考字数据优化模块
- 基于 ResNet-18 网络的与字体数据优化模块

模块的训练使用了古今结合的三大类型字体数据集，同时我们将三个模块整合为易用且专业的一站式字体创作工具，对专业外设实现了适配。模型的整体架构如下图所示：

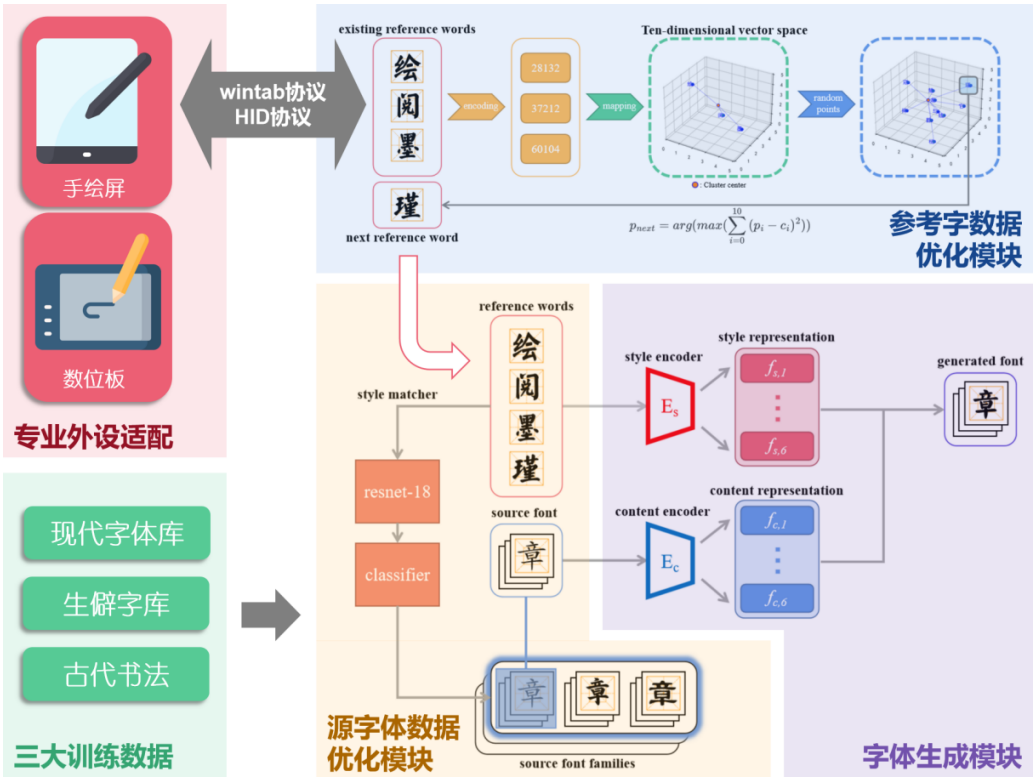


图 3.1: 模型总览

用户可以通过键鼠、触屏或是专业外设来绘制参考字。只需要在参考字数据优化模块的提示下，自由写入 10 个参考字，并且选择源字体数据优化模块所推荐的源字体，送入字体生成模块，得到最终生成的完整目标字体。除字体设计外，本应用还可用于现有字体对生僻字域的扩展和古书法的修复与存档。

3.2 核心生成模块——基于多头编码器

以往的的 AI 字体生成方法多是图像迁移方法（如 CycleGAN 等）的改进。对于图像风格迁移任务，通常涉及将一张图像的样式转移到另一张图像上，这两张图像可以来自不同的领域，比如一张风景照片和一张艺术绘画。在这种情况下，目标域通常是相对简单的，因为它只需要捕捉一些表面上的视觉风格，例如色调和纹理。此外，图像风格迁移任务通常有大量的训练数据可用，因此算法可以从中学习广泛的特征和图像表示。

相比之下，少样本字体生成任务面临的挑战更大。因为目标域是复杂的，生成新字体需要考虑大量的结构和几何形状的特征，例如字母间的间隔，笔画的粗细和形状，笔画之间的连通性等等。因此如果直接使用图像

迁移方法（仅仅使用生成对抗网络），并不能获得良好的字体生成效果，尤其是在面对复杂字形和非经典字形时存在较多的断笔、噪声等问题，如下图所示：

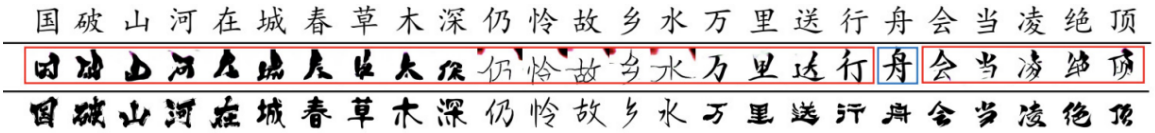


图 3.2: 直接应用传统 GAN 作为字体生成效果不佳

在生成网络的构建思路方面，我们从 Song Park 等人于 2021 年提出的 MX-Font 方法 [15] 中得到启发，在生成对抗网络的基础上加入了字体风格特征提取模块。该方法的主要工作原理是：使用多头编码器（multiple localized experts，多个局部专家）来抽取字形图像的特征，处理得到字形图像的内容特征和风格特征。然后将参考字的风格特征迁移和源字体的内容特征结合起来，送入生成器，得到最终的目标字体。

本模块可以分成两大部分：特征提取部分和风格迁移部分。特征提取部分由多头编码器、风格与组件特征分类器组成，风格迁移部分由字体生成器组成。模型架构如下图所示：

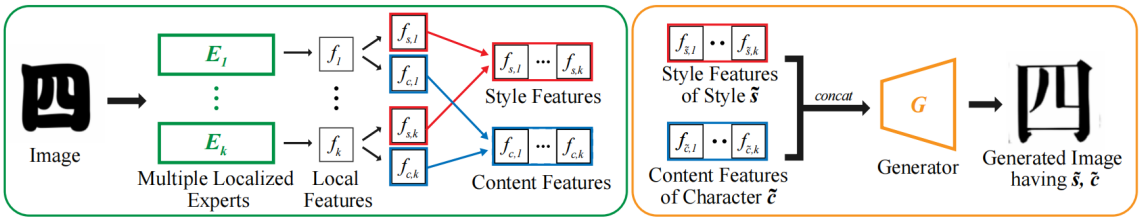


图 3.3: 字体生成模块总体结构

接下来分别从多头编码器，风格与组件特征分类器，生成器三个部分对该模块进行详细介绍。

3.2.1 多头编码器

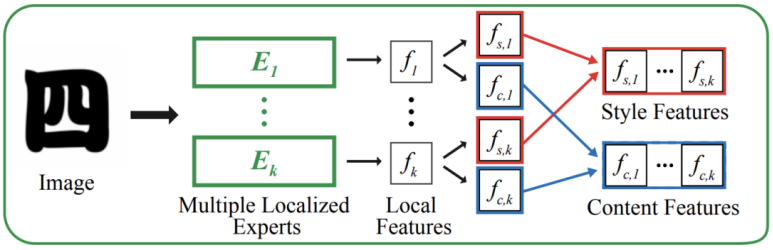


图 3.4: 多头编码器基本结构

如上图所示，是多头编码器的基本结构，编码器用于提取字形图像的特征，每个专家提取字形图像的局部特征。我们将其表示为一个组件分配问题，然后转化为加权二分图匹配（WBM）的问题，使用**匈牙利算法**解决。

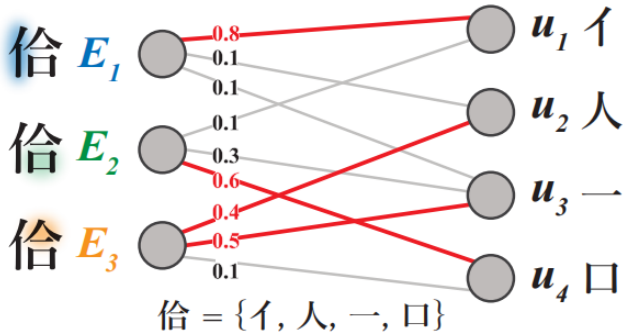


图 3.5: 组件分类问题

在一个二分图中，若每个左顶点最多与一个右顶点相连，则称这个二分图为二分图的最大匹配。匈牙利算法，也称为 Kuhn-Munkres 算法，就是为了求解这个最大匹配问题。匈牙利算法由两个基本部分组成：交替路的寻找和增广路的构建。其基本思想是，从左部顶点开始，不断寻找未被匹配的右部顶点，如果发现一个未被匹配的右部顶点，则将其与该左部顶点进行匹配。如果右部顶点已经被匹配了，那么就要尝试让与之匹配的左部顶点去匹配其他未被匹配的右部顶点，直到找到一个未被匹配的右部顶点或无法匹配为止。

匈牙利算法的时间复杂度为 $O(n^3)$，其中 n 是二分图中顶点的数量。虽然时间复杂度比较高，但是该算法在实践中表现出了良好的效果，被广泛应用于图像处理、计算机视觉、网络优化等领域。

第 i 个头，也就是专家 E_i 将字形图像 x 编码为局部特征 $f_i = E_i(x)$，将两个线性权重 $W_{i,c}$、$W_{i,s}$ 乘到 f_i 上，得到局部内容特征 $f_{c,i}$ 和局部风格特征 $f_{s,i}$。专家数 k 默认取 6。

3.2.2 风格与组件特征分类器

我们使用风格特征分类器 Cls_s 和组件特征分类器 Cls_u 来分别分类专家提取出的局部内容特征 $f_{c,i}$ 和局部风格特征 $f_{s,i}$。

如下图所示，k 个专家分别提取出来的局部风格特征 $f_{s,i}$ 组成了字体的整体风格特征，然后送入风格特征分类器 Cls_s 和组件特征分类器 Cls_u。Cls_s 对字体风格特征进行分类，而 Cls_u 则将字体风格特征预测为一个统一概率（等价于不进行任何预测）。类似的，Cls_u 对字体内容特征进行分类，而 Cls_s 则不进行任何预测。

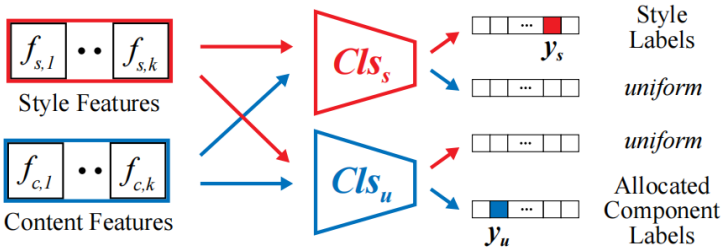


图 3.6: 特征分类器

将二分图匹配、特征提取和特征分类结合起来，可以将组件分配问题用如下形式表示：

$$\max_{w_{ij}\in\{0,1\},\, i=1,\ldots,k,\, j\in U_c} \sum_{i=1}^{k}\sum_{j\in U_c} w_{ij}P_{ij} \tag{3.1}$$

$$\text{s.t.} \quad \sum_{i=1}^{k} w_{ij} \geq 1 \quad for \quad \forall j, \quad \sum_{j\in U_c} w_{ij} \geq 1 \quad for \quad \forall i \tag{3.2}$$

$$\sum_{i=1}^{k}\sum_{j\in U_c} w_{ij} = \max(k, m) \tag{3.3}$$

其中 p_{ij} 是组件 j 的置信度，$U_i = [u_1^c, \ldots, u_m^c]$ 为给定字符 c 的组件标签，m 为组件数量。使用二元分配量 w_{ij}，当其值为 1 时表示组件 j 分配给专家 E_i，否则不分配。优化 w_{ij} 来最大化选中预测值的和，总分配数为 $\max(k, m)$。$\sum_{i=1}^{k} w_{ij} \geq 1 \quad for \quad \forall j$ 表示每个组件至少会分配给一个专家，$\sum_{j\in U_c} w_{ij} \geq 1 \quad for \quad \forall i$ 表示每个专家至少会分配到一个组件，$\sum_{i=1}^{k}\sum_{j\in U_c} w_{ij} = \max(k, m)$ 表示选择中的边数不超过 $\max(k, m)$。该公式可由匈牙利算法在多项式时间 $O((m+k)^3)$ 内求解。

损失函数的目的是当专家 i 被分配到组件 j 时，使 $Cls_u(f_{c_i})$ 接近组件 j：

$$L_{\text{cls},c,i}(f_{c,i}, U_c) = \sum_{j\in U_c} w_{ij}\,\text{CE}(\text{Cls}_u(f_{c,i}), j)$$

3.2.3 生成器 G

多头编码器和特征分类器分类训练完成后，就可以联合各专家对原字形以及目标字形的内容、风格特征来产生字形 $\widetilde{\mathrm{x}}$：

$$\widetilde{\mathrm{x}} = \mathrm{G}((\mathrm{f}_{\mathrm{s},1} \circ \mathrm{f}_{\mathrm{c},1}), ..., (\mathrm{f}_{\mathrm{s,k}} \circ \mathrm{f}_{\mathrm{c,k}}))$$

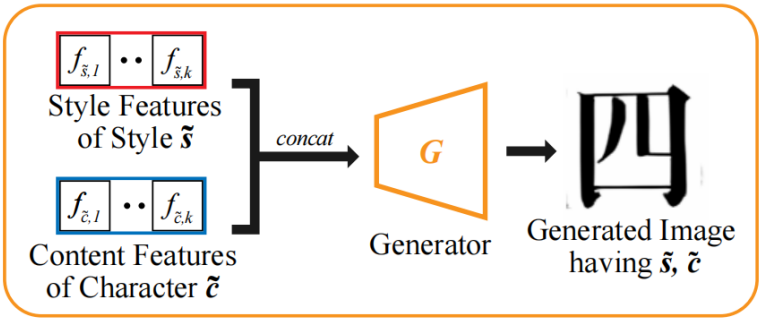


图 3.7: 生成器用于生成目标字形

该部分的设计采用的是生成对抗网络的思想。除了字体生成器外，还引入了一个分类模块 D（包含两个判别器 D_s 和 D_c 分别预测样式标签 y_s 和内容标签 y_c）。引入 hinge 生成对抗损失 L_{adv}，目的是让生成器和鉴别器进行博弈，最终让鉴别器难以区分生成器生成的图片和真实的图片，从而提高生成器生成图片的质量；特征匹配损失 L_{fm}，目的是让生成器生成的图片尽可能与源字形以及目标字形的内容、风格特征匹配；像素级别重建损失 L_{recon}，目的是让生成器生成的图片拥有视觉高质量。损失函数如下，L_D 表示判别器的损失函数，L_G 表示生成器的损失函数：

$$\mathrm{L_D} = \mathrm{L}^{\mathrm{D}}_{\mathrm{gdv}}$$

$$\mathrm{L_G} = \mathrm{L}_{\sigma\mathrm{dy}} + \lambda_{\mathrm{recon}}\mathrm{L}_{\mathrm{recon}} + \mathrm{L}_{\mathrm{fm}}$$

本模块在上层软件界面的部署如下图所示：

图 3.8: 少样本字体生成模块在软件界面上的部署

3.3 参考字数据优化模块——基于四角编码映射

目标字体的风格依赖于所提供的少量参考字，所以为了最大化地优化模型效果，我们希望用户能够尽量地提供带有更多样化字形信息的参考字组，这也是目前 AI 生成字体所面临的一大挑战。在实验过程中，我们发现对于目前的风格迁移网络，至少需要按要求提供 200~300 个参考字，才能达到较为优良的生成效果，大量参考字的书写使得制作流程繁琐，降低了用户的使用体验，也违背了使用 AI 生成字体简化繁琐的字体设计流程的初衷。

3.3.1 参考字列表的动态更新

针对该问题，本软件提出了基于汉字四角编码的参考字推荐算法。软件不指定用户需要提供的参考字，而是交由用户自行选择。每当用户绘制完一个参考字，或是改动已有的参考字，将自动进行 OCR 汉字识别，并且动态更新当前的参考字列表。同时基于已有的参考字列表调用基于四角编码的最大化距离向量算法，动态更新下一个推荐的参考字。

对用户提供的参考字图片进行动态 OCR 的流程如下图所示，当参考字没达到其阈值时，将会不断对当前的参考字列表进行动态更新，为用户推荐下一个书写的汉字。[4]

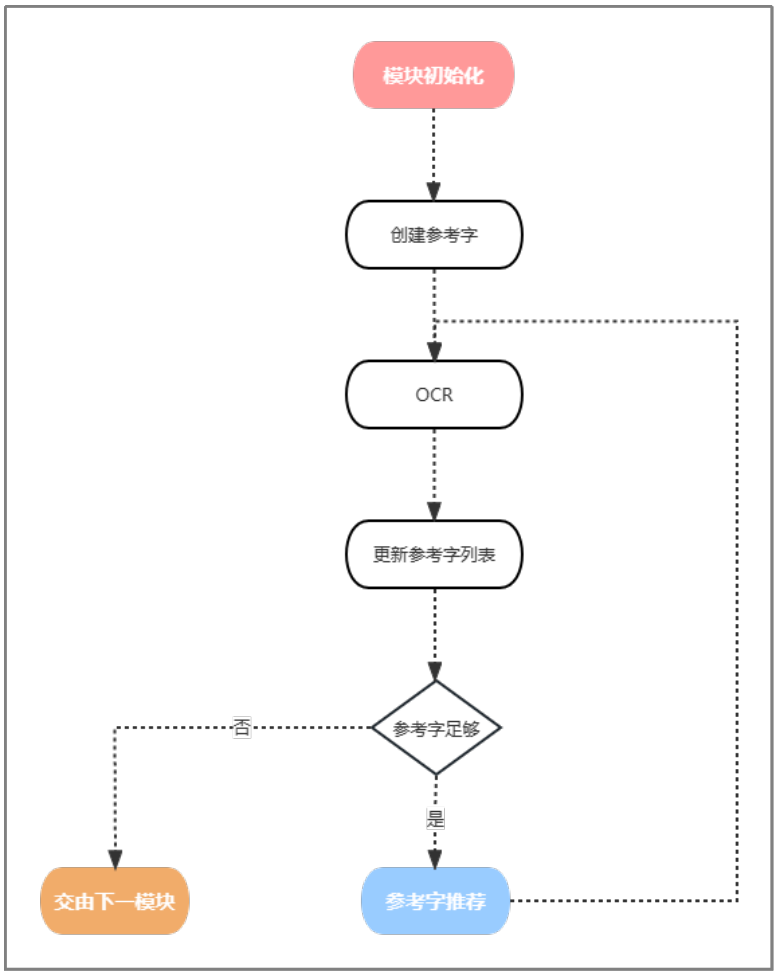


图 3.9: 参考字列表动态更新流程

3.3.2 基于四角编码的最大化距离向量算法

在获得了当前的参考字列表之后，本模块将向用户推荐一个与当前参考字的字形区别尽可能大的常用字。本算法使用汉字的四角编码来表示汉字的字形。

四角号码检字法 [17] 是一种基于汉字方块字形，对汉字角部件结构进行编码的检字方法。它把汉字笔形分为十类：头、横、垂、点、叉、插、方、角、八、小，分别用数字 0～9 表示。每个字四个角的笔形按其位置左上、右上、左下、右下的顺序取号，并用（四角 + 附角）5 个阿拉伯数字表示。汉字四角码的编码规则如下图所示：

笔名		号码	笔形	字例
复笔	头	0	亠	主病广言
单笔	横	1	一	天土
			𠂇 乚 𠃌 𠂆	活培织兄凤
	垂	2	丨	旧山
			𠂉 丿 亅	千顺力则
	点	3	丶	宝社军外去亦
			𠂉 乀	造瓜
复笔	叉	4	十	古草
			𠂇 乂 犭 𠂆	对式皮猪
	插	5	丰	青本
			扌 戈 丈 𢦏 丰	打戈史泰申
	方	6	口 冂	另扣国甲由曲
			冂 冂	目四
	角	7	𠂉 𠂉 乚 𠂉	刀写亡表
			𠂉 𠂉 亅	阳兵雪
	八	8	八 𠂆	分共
			人 入 𠂆 𠂉 𠂉	余汆央羊午
	小	9	小	尖宗
			忄 𠂆 𠂉 灬	快木录当兴組

图 3.10: 汉字四角码的编码规则

对于含有相似字形部件的汉字，其四角码的数字分布往往类似，如下图所示，拆”、“折”、“端”三个字的部件拆分及四角编码，可以看到“2”代表的是横或或竖结构笔形而“5”、“0”表示的是提手旁。含有相同笔形汉字四角码的数字分布相似。

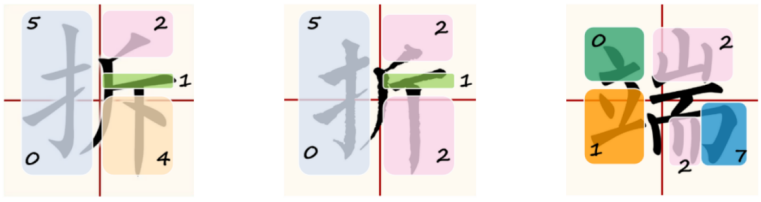


图 3.11: 含有相似字形部件的汉字，其四角码的数字分布往往类似

基于以上原理，本算法将汉字的四角编码依据其中 0 9 出现的次数按照下式的方法来映射到 10 维向量空间中：

$$\mathrm{D[i]} = \mathrm{count(F, i)}$$

其中 F 为汉字的四角码，$count(F, i)$ 表示 i 在 F 中出现的次数，D 为该汉字映射为的 10 维向量。基于以上

论断，两个汉字 Z_1 与 Z_2 对应的向量 D_1 与 D_2 之间的欧几里得距离可用于衡量两个汉字所包含的笔形相似度，欧几里得距离越短者说明两个汉字具有越相似的笔形构成。

对于已经提供的参考字列表，首先计算当前所有参考字在 10 维向量空间中的聚类中心 C：

$$\mathrm{C} = \frac{\sum_{j=1}^{N} \mathrm{D_j}}{\mathrm{N}}$$

最后在向量空间中随机选择 10 个常用字作为候选字，选取其在向量空间中距离 C 的欧几里得距离最长者 P 作为推荐的下一字：

$$\mathrm{P_{next}} = \arg(\max(\sum_{\mathrm{i}=0}^{10}(\mathrm{P_i} - \mathrm{C_i})^2))$$

下图展示了该算法完整执行一次的流程，假设当前的参考字列表为“绘”、“阅”、“墨”，首先将其进行四角编码，得到“28132”、“37212”、“60104”，然后三个字对应的四角码分别映射到 10 维向量空间中，计算出聚类中心（图中橙点），并在空间中选取 10 个候选字，选择与聚类中心欧几里得距离最大者“瑾”作为下一个推荐的参考字。

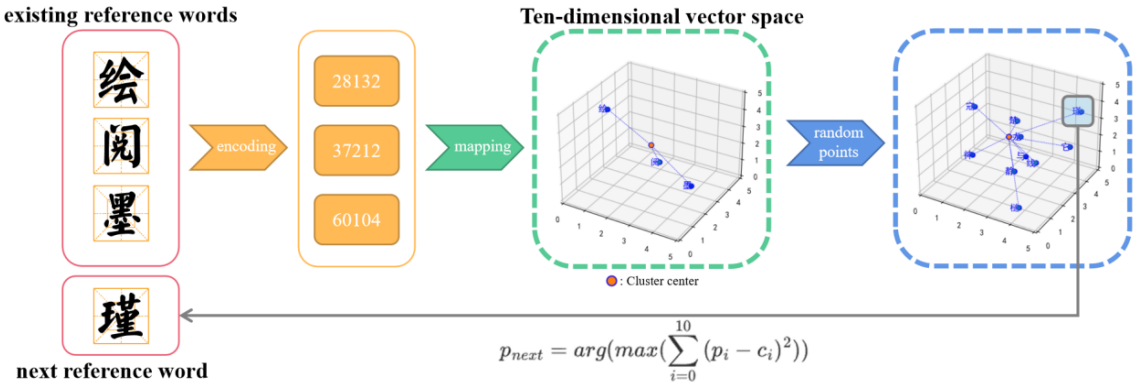


图 3.12: 参考字推荐算法执行流程

同时，为了出于对用户自由的考虑，如果用户想更换推荐参考字，可以重新随机取点，当用户没有按照参考字推荐进行书写时，参考字列表和聚类中心也能基于动态 OCR 机制做出动态更新。下图分别展示了在参考字列表长度分别为 1、3、5、7 时，连续调用该算法所生成的参考字列表：

图 3.13: 在不同规模的参考字列表下连续执行本算法得到的推荐字序列

观察上图可以看到本算法可以根据用户输入的推荐字序列动态更新聚类中心，基于字形多样性最大化原则产生不同的推荐字序列。

3.4 源字体数据优化模块——基于 ResNet-18 网络

模型将参考字的风格特征迁移到源字体的结构上，当参考字与源字体的风格差异较大时，所生成的目标字体便会同时带有两种字形的风格，该现象被称为风格杂糅，如下图所示，当发生风格杂糅时，所生成的字体实际上是不可用的。在实际实验之中，往往需要人工进行多次尝试，才能将字体生成风格杂糅的现象最小化，如何在算法层面抽取并识别参考字的风格，并迅速为其匹配出风格最相似的源字体，也是目前 AI 生成字体面临的一大挑战。

下图分别展示了使用与参考字不相似的源字体和相似的源字体所生成的效果。可以看到，当使用与参考字风格不相似的字体作为源字体时，所生成的字体会发生风格杂糅，而当使用与参考字风格相似的字体作为源字体时，所生成的效果相对更好。

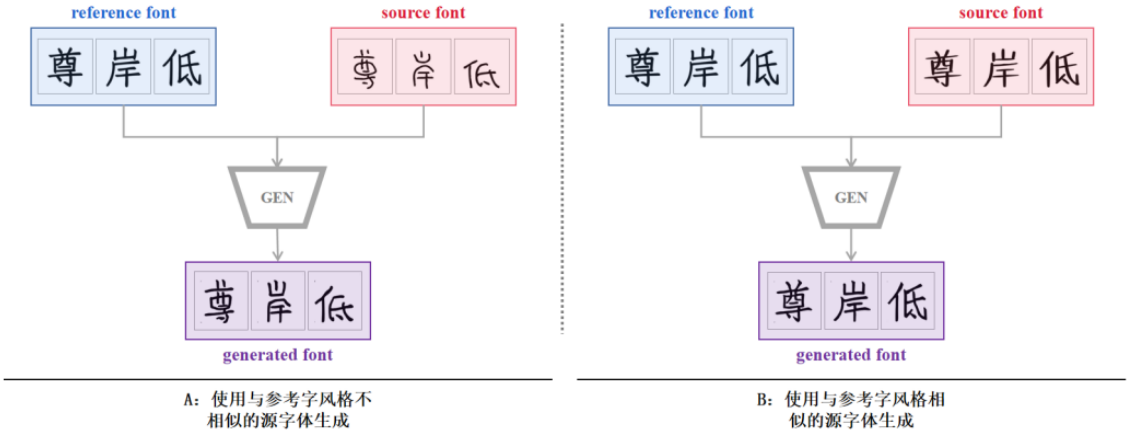


图 3.14: 风格杂糅现象的示意

为了从源字体数据的角度优化生成模型的表现，我们设计了源字体智能匹配模块，对于用户提供的参考字，将其送入一个基于 ResNet 网络的风格匹配器中，并选择与参考字风格最相近的字体作为源字体；以达成最小化风格杂糅的目的。

3.4.1 源字体匹配模块原理与构建

上文提到的风格匹配器旨在提取汉字的风格特征，以达到为参考字体匹配风格最接近的字体作为源字体。我们将其作为一个字体风格分类任务，使用和字体生成器相同的字体数据集作为训练集（下一节详细介绍），采取 ResNet-18 作为分类网络，提取字体的风格特征并判断字体类别。

使用 ResNet-18 作为分类网络对字体进行分类。ResNet-18[18] 是深度残差神经网络（Deep Residual Neural Network）的一种，由微软研究院于 2015 年提出。它由 18 层卷积神经网络组成。

相较于传统的卷积神经网络，ResNet-18 具有两个特点：残差结构和跨层链接。残差结构是指在网络中间层的输出添加了跨层链接（shortcut connection），使得信息可以更直接地传递到后续层，防止梯度消失和梯度爆炸，从而加速网络训练和提高准确性。跨层链接也被称为“恒等映射”，它可以通过将输入直接传递到输出层来避免信息的丢失和损失，保证信息的完整性。

ResNet-18 中包含多个残差块，每个残差块包含两个卷积层和跨层链接。它的输入是一个 224x224 大小的 RGB 图像，经过一系列卷积层、池化层、全连接层等操作，最终输出图像的类别。目前，ResNet-18 被广泛应用于图像分类、目标检测、语义分割等领域，其在 ImageNet[2] 数据集上的表现也得到了广泛认可。风格匹配器的网络结构如下图所示：

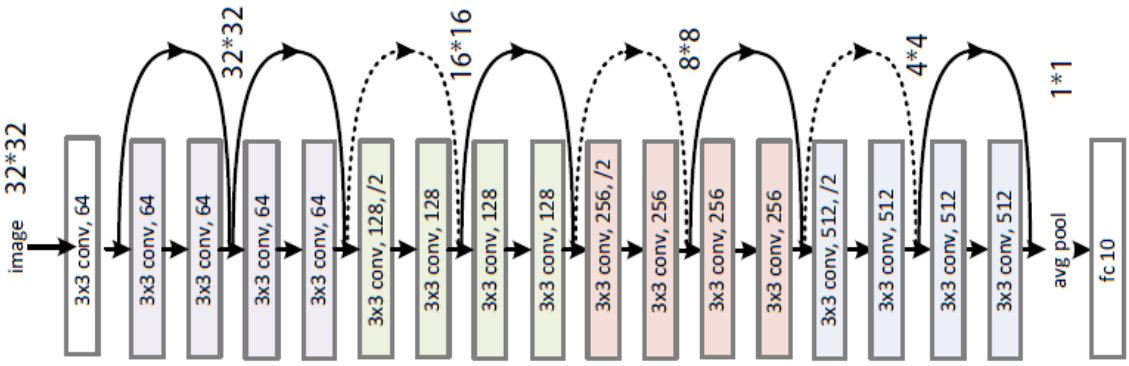


图 3.15: 风格匹配器网络结构

下图展示了 ResNet 网络的具体细节：

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

图 3.16: ResNet 网络的具体细节

当参考字列表已经确定，在进行最终的目标字体生成前，参考字将被送入风格匹配器，模块对其字体风格进行分类。确定字体风格后，依次计算该字体与该类别中的 10 种字体的相似度，然后返回与其最相似的 3 4 种字体作为候选源字体，供用户选择，损失函数如下图所示，其中 $\mathrm{d}(\mathrm{f_i}, \mathrm{f_{target}})$ 表示参考字特征和候选字体特征的距离。参考字的选择算法可以用下式表示：

$$\mathrm{Font}_{ref} = argmin_{f_i \in F} d(f_i, f_{target})$$

3.4.2 风格匹配器部署与应用

模块的执行示意图和效果展示如下图所示：当得到参考字列表时，将其送入基于 ResNet-18 构建的网络，预测出参考字对应的字体风格分类，并在其中选择最相似的 3 4 个字体当做源字体候选。

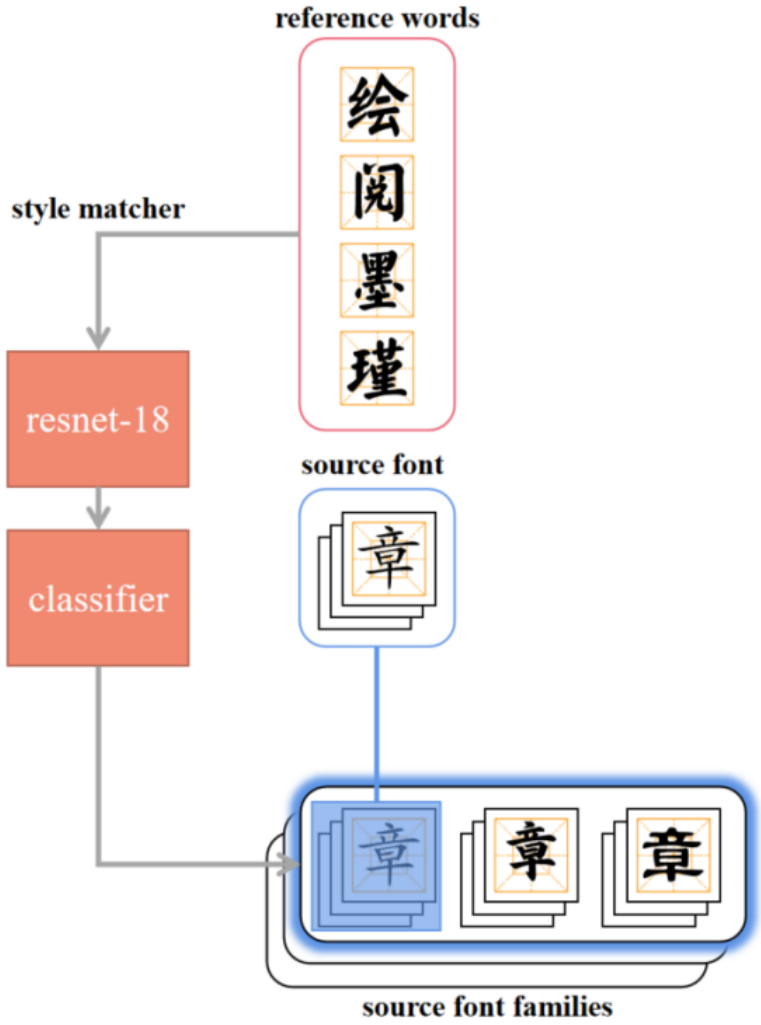


图 3.17: 风格匹配器工作示意图

本模块在上层软件界面的部署如下图所示，软件在得到候选字体后提示用户选择。

图 3.18: 源字体数据优化模块在软件界面上的部署

3.5 模型训练数据——古今结合助力模型优化

3.5.1 思路概览

为了进一步增强字体生成模型在复杂结构字形上的生成表现，仅仅从模型架构上进行改动是远远不够的。所以我们从数据的角度，基于大数据思想，构建了现代字体库 + 生僻字库 + 古书法字库三种类型共数万字的汉字字形数据集，用于优化生成模型的表现，以及构建两个数据优化模块。

字形大数据信息的获取和处理流程如下图所示：

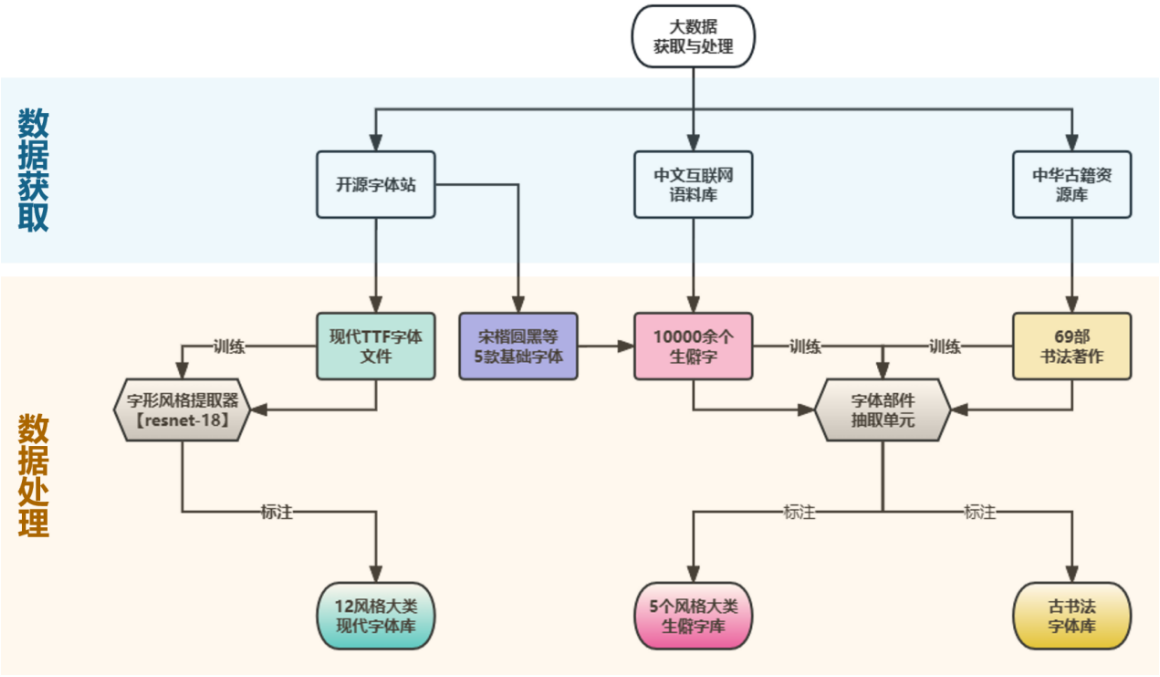


图 3.19: 字形大数据信息的获取和处理总体架构

接下来我们将从这三大数据的角度分别详说明本团队提出的思路：

3.5.2 现代字体库

现代字体库是传统的 AI 字体生成模型的基础，每一种字体以 ttf 文件的形式组织，如下图所示，ttf 文件实际上指明了字体中不同汉字的字形图片，包括不同尺寸的字形。我们将 ttf 文件拆分为一张张 png 图的形式，尺寸固定 128*128，色深为 8bit，以此作为模型的第一大数据集。

图 3.20: ttf 文件（以微软雅黑为例）

我们设置了爬虫，从字由 [3]、100font[4]、求字体网 [5]、字魂 [6] 等十余个开源字体网站爬取现代字体，去重之后，最终得到 500 余种字体。

接下来为了使这部分数据能够用于源字体数据优化模块的构建，我们还按照字体风格将这 500 种字体分为 12 大类，经过数据统计和分析，我们概括出分类效果和分类依据，如下图和下表所示：

图 3.21: 12 个风格大类的示意

风格	风格特点
楷书	结构方正，起笔、运笔和收笔有变化，折笔不停
宋体	笔画有粗细变化，而且一般是横细竖粗，末端有装饰部分，点、撇、捺、钩等笔画有尖端
行书	包括行楷和草书，大小错落有致，点画以露锋入纸的写法居多，以勾、挑、牵丝来加强点与画的呼应
标题字	与黑体类似，但是相比比宋体，其笔画结构会更具特点；字形结构也更灵活多变
篆刻	字形多呈宽扁，横画长而竖画短，有“蚕头燕尾”、“一波三折”的特点
空心	字体笔画无填充，仅保留笔画的轮廓
黑体	又称方体或哥特体，没有衬线装饰，字形端正，笔画横平竖直，笔迹全部一样粗细
手写体	人的手写体，结构较为随意，笔画粗细/曲度较为参差不齐
毛笔字	使用毛笔来写的书法字体，其结构较为随意，但是起笔收笔都有笔锋
圆体	由黑体演变而来，拐角处、笔画末端为圆弧状
卡通体	和圆体一样有笔画圆润的特点，但是相比起来结构更随意
海报体	同时具有黑体的笔迹等粗、圆体的拐角圆润和手写体的结构随意的特点

表 3.1: 12 个风格大类的分类依据

这 12 大类 500 种字体不仅能够用于字体生成模型的训练和优化，也是源字体数据优化模型构建的基础。

3.5.3 生僻字库

根据 1994 年冷玉龙等的《中华字海》，生僻字多达 85000 字，其中约有 10000 字是在人名、地名中常见的生僻字，一方面因为其字形复杂独特，另一方面因为现存的数字化字体的汉字数目都在 5000 字左右，所以使用一般常用的数据集所训练的模型，在生僻字的生成效果上表现并不好。

为了能够充分实现在复杂字形和非经典字形上的生成效果，我们独创性地引入了生僻字库作为模型的第二大训练集。我们首先爬取了 10 万余字的中文语料库，统计其中按照时间加权词频在前 10% 的生僻字构建生僻字库。

因为对于生僻字，我们无法得到其具体的组件序列，但这确实模型训练所需要的，所以我们设计了基于 ResNet18 编码器和 attention 机制的解码器的组件序列标注模块，用于对无标注的生僻字生成其标注。

最终我们在数据集之中引入了楷体、黑体、等线、幼圆、宋体五个基本字体的 10000 余个生僻字，作为生僻字库。用以优化模型在复杂字形和非经典字形上的表现。

3.5.4 古书法字库

以上提到的现代字体库和生僻字库，都是基于现代数字化字体建立的，而中国古代书法之中也存在着非常多样且经典的汉字与字形风格，一些古文字的部件同样可以为模型在非经典字形汉字的生成效果上起到促进作用。

在现代字体设计之中，也经常会利用一些名家名篇的书法风格；同时，将古书法字库加入数据集中，也能优化模型对古籍古碑上文字风格的提取能力，使得工具能够应用于古籍修复等功能之中。

所以我们从中华古籍资源库 [7] 中，爬取包括《玄秘塔碑》、《兰亭集序》在内的 69 部经典书法著作，如下图所示。其中包括篆、隶、行、楷等多种风格，从中选取了 5000 个清晰的汉字，截取、去噪之后构建起古书法字库。

图 3.22: 所爬取的古书法著作

该字库同样使用上一节提到的字体部件抽取单元其部件组成序列进行标注，作为最终数据集中的第三部分。实验证明引入了生僻字和古书法字库，对模型的生成表现起到了 17% 以上的优化作用。

3.6 专业外设支持——一站式解决方案

3.6.1 硬件系统总体架构

本系统的硬件支持包括两个部分：在绘图阶段对专业设计外设的支持和在生成阶段对模型运行环境的支持，硬件系统总体架构图如下：

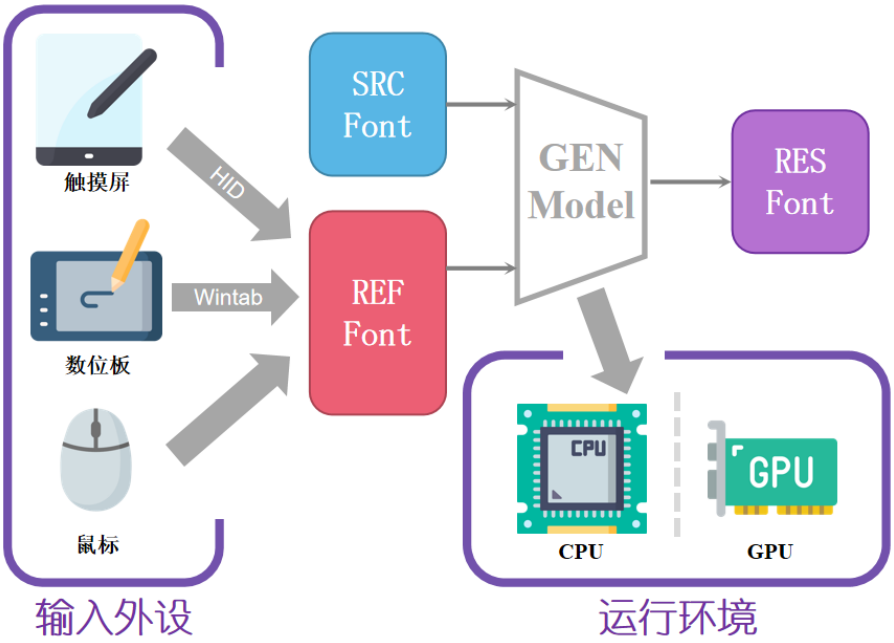


图 3.23: 所爬取的古书法著作

3.6.2 绘图输入——手写笔和触摸屏

在参考字设计方面，本软件支持三种绘图输入方法：

- 触摸屏：如市面上常见的 Surface pen+Surface 的设备组合，用户直接使用手写笔在屏幕上进行绘制。
- 数位板：在专业绘画创作领域常见的外设，用户使用专业的压感笔在数位板上作画。
- 鼠标：使用光标控制笔在画布上的位置

触摸屏和数位板的输入分别基于 HID 协议和 Wintab 协议实现，如下表所示：

协议	支持设备	功能	调用接口
HID	触摸屏	捕获手指和手写笔在屏幕上的压力和轨迹信息，以获取笔触和笔尖位置	hidapi
Wintab	数位板	捕获手写笔在配套数位板上的压力和轨迹信息，以获取笔触和笔尖位置	Wintab API

表 3.2: 硬件协议与接口

在软件实现层面，当打开了画布对应的线程之后，线程内将会开启对触摸屏和数位板外设对应的接口进行循环侦听，以捕获用户的绘图输入事件，程序的执行流程如下图所示：

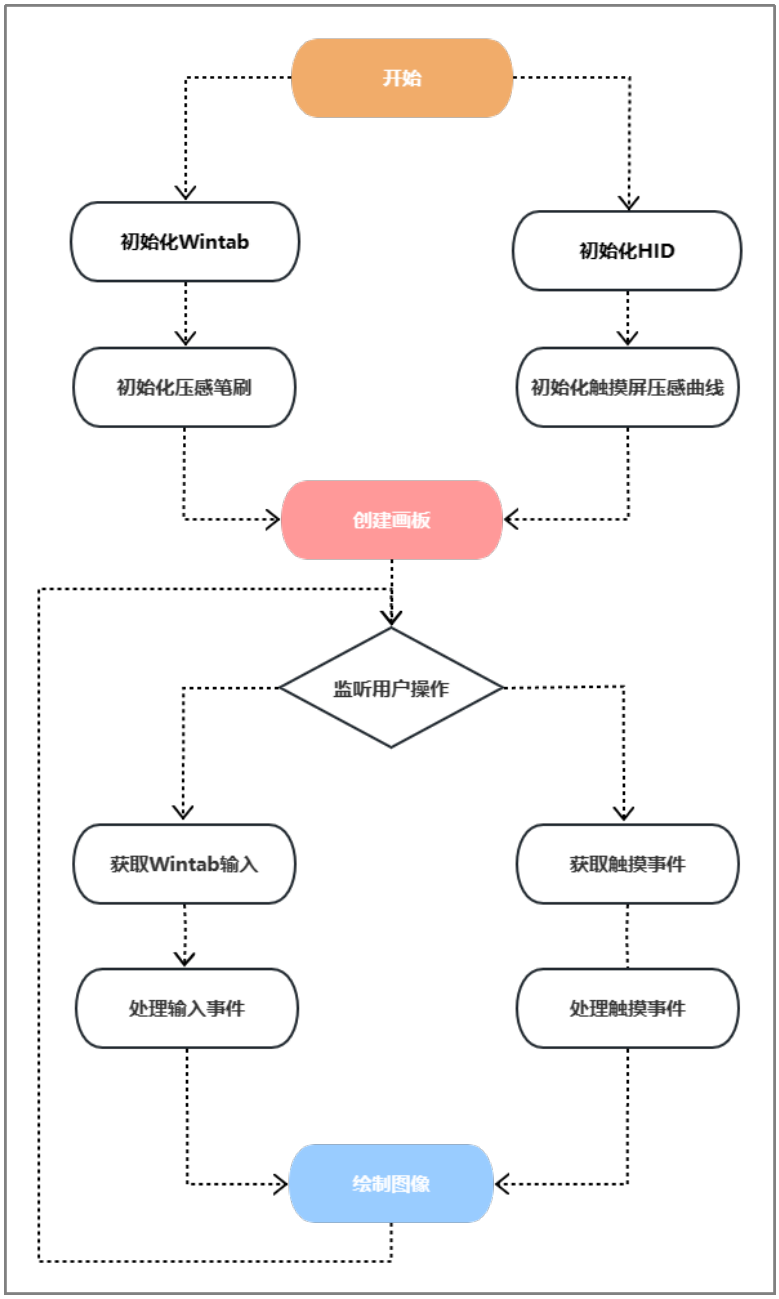


图 3.24: 捕获用户绘图输入执行流程

3.6.3 字体生成——GPU 设备

在模型的运行阶段，本项目对不同配置的硬件环境都做出了适配。当用户主机中配置了英伟达 GPU 和 cuda 环境时，我们会将模型迁移到显存中运行。

第 4 章 系统实现

4.1 技术栈总览

本项目部署于 Windows x64 环境下，在工程实现方面分为软件设计、三大模块的部署、硬件设备三部分：

在上层设计部分，本项目基于 pyGame 和 tkinter 实现了前端可视化界面，软件主体逻辑和三大模块接口的调用由 python 多线程框架进行实现。

对于参考字数据优化、源字体数据优化和字体生成三大模块，在参考字的识别方面我们调用了百度提供的 OCR api，而参考字的选择、源字体匹配和字体生成模型则基于 pytorch 框架进行实现。

在硬件设备的支持方面，本项目使用 Wintab 协议和 HID 协议实现了对数位板和触控屏的支持，同时 pytorch 模型在 cuda 架构之上开发，实现了对 GPU 设备的支持。

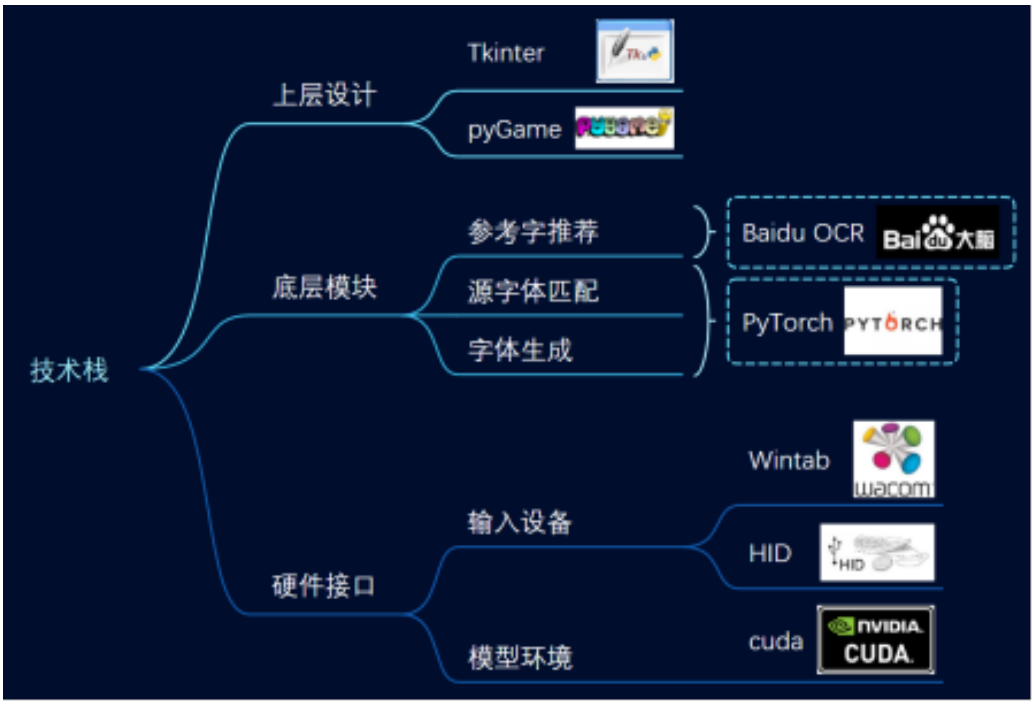


图 4.1: 本项目工程实现的技术栈

接下来将从软件设计、三大模块的部署和硬件设备接入三个部分对系统的实现作出详细介绍。

4.2 软件上层设计

4.2.1 程序主体分析

本软件的逻辑主体使用 python 编程语言实现，为了能在同一个程序界面内实现包括参考字绘制、参考字数据优化、源字体数据优化、目标字体生成及展示几个子功能，主体程序将基于多线程 python 编程框架来开发。以下是程序主体执行流程图。

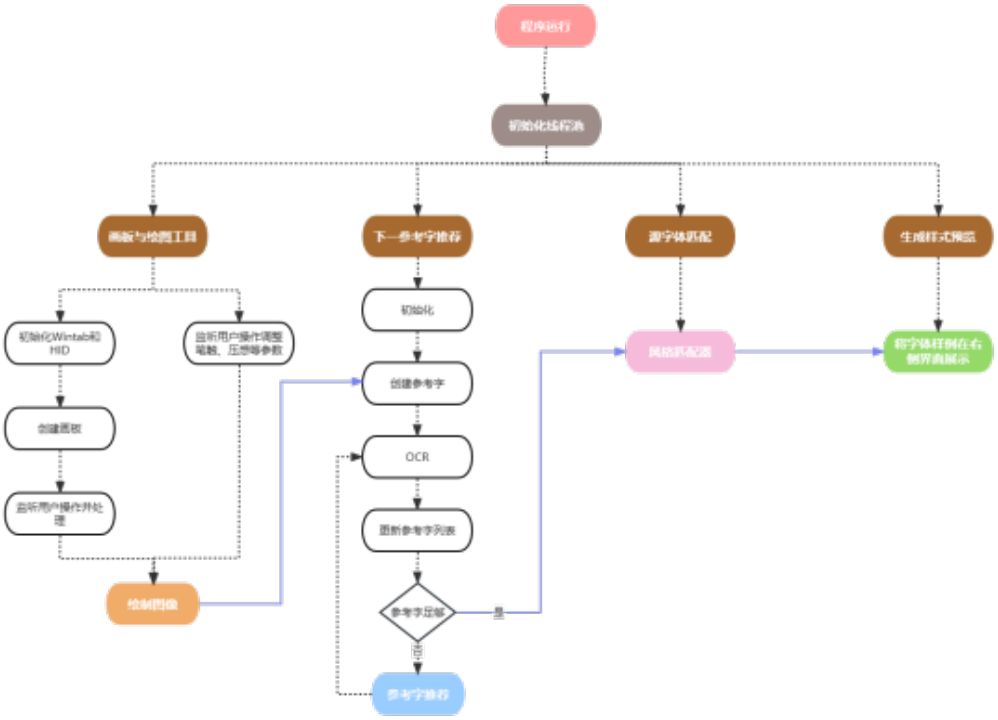


图 4.2: 程序主体执行流程图

如上图所示，当软件启动时，将打开四个线程，分别对应画板与绘图工具、参考字推荐、源字体风格匹配和生成样式预览四个主要功能。四个线程在程序运行时并行执行，并使用 quene 库提供的接口来实现线程间通信。

- 画板与绘图工具线程负责连接硬件外设端口，构建起基于 canvas 的画布来实现参考字的绘制功能。当线程打开时，程序将不断循环侦听用户的绘图操作（例如手写笔或光标的轨迹变化以及压感变化等），同时当用户修改工具栏或者笔触栏参数时，也能及时反馈到 canvas 之上。
- 每当系统侦测到用户超过一定时间阈值没有绘图操作后，将自动唤醒参考字推荐线程。首先调用百度提供的手写文字识别 API 来对用户新绘制的参考字图像进行 OCR，以更新对应的参考字列表，然后将参考字列表送入参考字数据优化模块，计算出能最大化参考字多样性的下一字推荐，并显示在界面上。
- 当已经提供的参考字数满足了所要求的数目时，将自动唤醒源字体匹配线程，将当前的参考字列表送入源字体数据优化模块，该模块返回源字体库中匹配到的四种风格最相近字体，以预览图的形式在界面上呈现，并可供用户选择。
- 在用户选定源字体之后，将用户提供的参考字列表和所选择的源字体送入字体生成模块，等待生成模型产生最终的目标字体及其预览图。

4.2.2 用户界面设计

本软件的应用界面充当上述提到的四个功能线程的交互接口，分为四个部分：

画板与绘图工具，位于界面的左半部分，左边的工具栏集合了一般绘图软件常用的放大缩小、压感粗细调节等设计功能，而上方的笔触一栏则可供用户快速地切换如毛笔、钢笔等不同笔触。

下一参考字推荐，位于界面的右下方，每当用户确定完成一个参考字之后，都会调用参考字数据优化模块中的最大化向量距离算法，由右下角的卡通 AI 助手来向用户提出推荐建议，具有人性化且满足轻量化设计的需求。

源字体匹配，位于软件界面的中间一栏，当用户提交了所有的参考字之后，将会调用源字体数据优化模块中的字形分类匹配算法，在 200 余个源字体中选择和当前参考字字形最相近的 3 4 种字体供用户选择。

预览结果的生成与导出，位于界面的右上方。当字体生成模型生成完成后，将在该栏中展示 8 个生成的示

例汉字。用户查看预览效果觉得满意后，可以在右上角“文件”处选择将生成结果导出，支持 png、jpg、ttf 文件等多种形式。

以下将结合应用截图，按照软件的使用流程对用户的界面依次进行介绍：

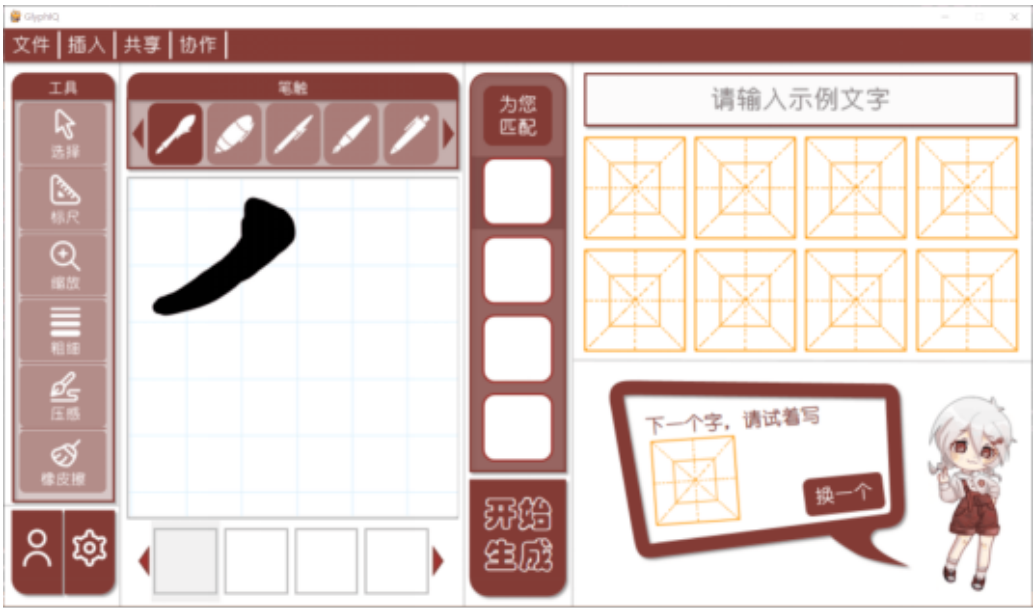


图 4.3: 绘制参考字

上图展示的是参考字绘制过程的软件示意图，用户可以在左边的绘画框处，使用不同粗细、压感曲线、笔触的笔，在像素规模为 1024*1024 的画布上进行作画，并且调整参考字的笔画细节。

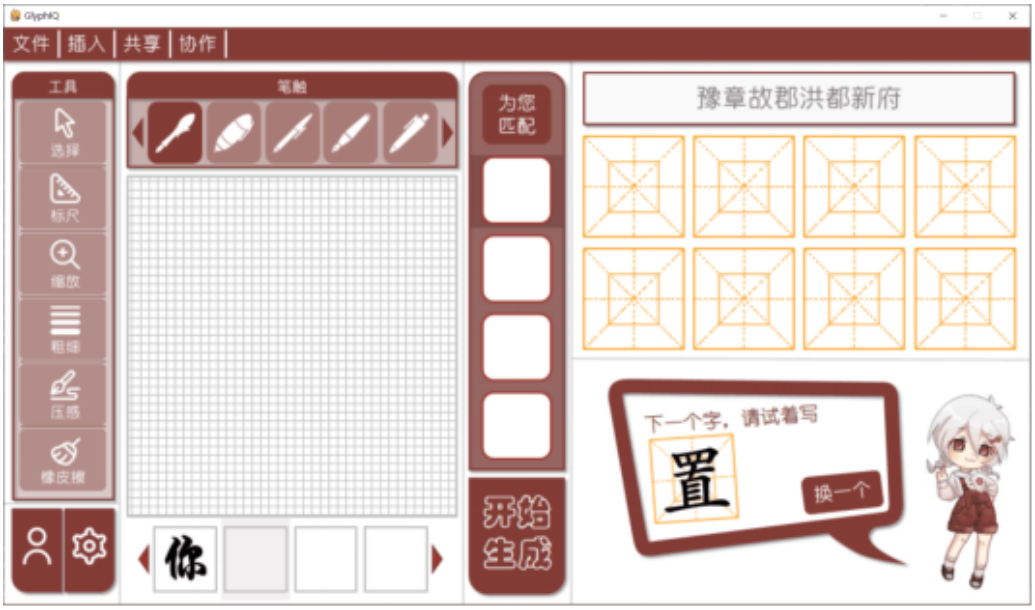


图 4.4: 推荐下一个参考字

上图展示的是参考字数据优化模块的工作效果，当用户已经写好了一个参考字，如下图中的“你”字，此时下方的卡通 AI 助手将根据已经写了的字的字形为用户推荐下一个字。

图 4.5: 匹配最相似的源字体

上图展示的是源字体数据优化模块的工作效果，当提供的参考字个数满足了生成的需要，智能助手将会提示用户选择源字体，此处选择了第一个字体作为源字体，点击开始生成，软件将会显示“正在生成”，等待数分钟后（由用户的主机配置决定），便可以展示出预览效果。

图 4.6: 预览生成结果与导出

图 4.7: 保存为.ttf 文件

上方两图展示了生成完成和字体文件导出的流程，当查看预览效果觉得满意之后，用户可以在右上角“文件”处选择将生成结果导出，可选择导出为 png 图片集、ttf 文件的形式，或者保存当前的设计结果。

除此之外，我们也对应用的功能进行了拓展，使用少样本生成的功能，用于字体生僻字域扩展和古书法保护：

图 4.8: 导入待扩展的字体 ttf 文件

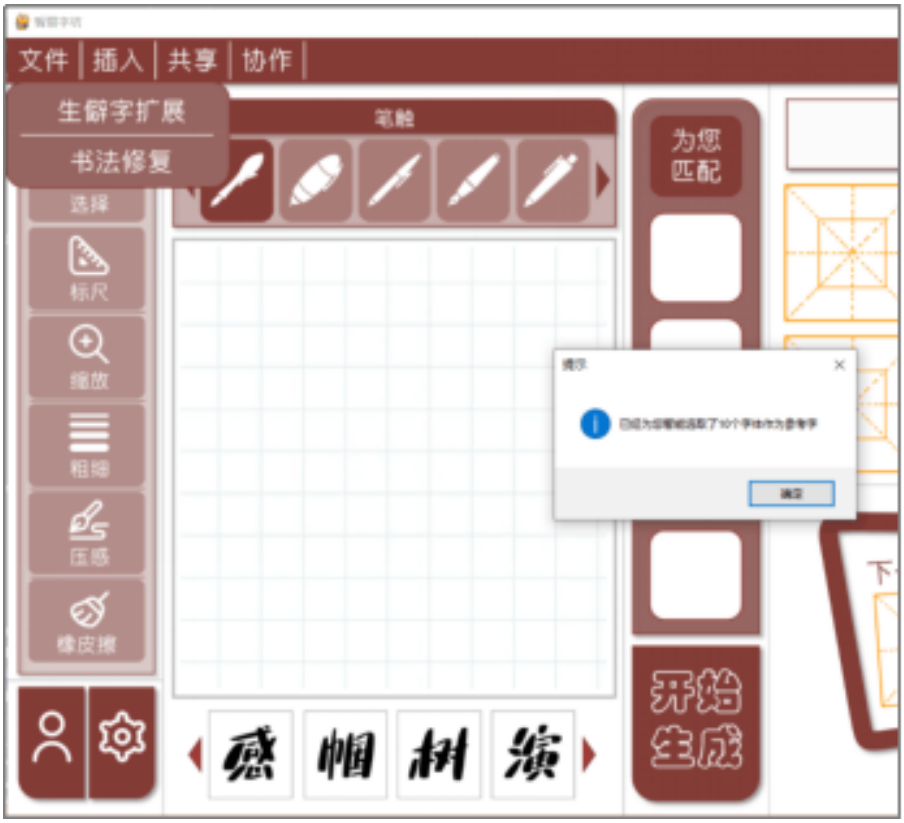


图 4.9: 自动在其中选择 10 个参考字

如上图所示，对于已有字体，本软件同样可以学习其字形风格，然后将风格迁移到这款字体所不支持的生僻字域之中。

图 4.10: 导入书法图片

同样地，本软件也可以用于古代书法风格提取和修复，将 png 图片形式的书法汉字图片导入作为提供的少样本参考字，从而提取出书法作品的风格，还原出对应的字体。

4.3 模块部署

4.3.1 基于多头编码器的字体生成模块

字体生成模块的数据集包含三部分：现代字体库、生僻字数据集、古文字数据集。

现代字体库包含了从字由、100font、求字体网、字魂等十余个开源字体网站爬取的 12 种风格共 510 个不同的开源字体：

图 4.11: 12 大类字体风格示例

同时我们对这 510 种开源字体所包含的汉字数量进行统计，统计结果如下图所示：

图 4.12: 几种常见字体包含的汉字数量统计

观察可知，宋体、雅黑、等线、楷体、黑体这 5 种字体所包含的汉字数量最多，因此我们从这 5 种字体中精选了 1 万个生僻字，这些生僻字在这 5 种字体中都存在，以此作为我们的生僻字数据集。

此外，我们从中华古籍资源库爬取了 37 位作者的 69 部古籍，并从中截取了 5000 张清晰的古文字体图片，以此作为我们的古文字数据集。

图 4.13: 中华古籍资源库

综上所述，用于模型训练的数据集共有 547 种字体，包含常用字、生僻字、古文字。同时我们手工标注了数据集的风格标签和组件标签。风格标签如下表所示：

表 4.1: 数据集的风格标签

数据集	风格标签
现代字体库	字体名称（510 种）
生僻字数据集	字体名称（5 种）
古文字数据集	作者名称（37 种）

汉字的组件标签是指以笔画、字形等基本组成单位分解该汉字得到的多个文字。例如“字”可以拆分为“宀”和“子”。为了得到汉字的组件标签，我们使用了 Github 中的 chaizi 项目所提供的汉字拆字字典，其中收录了 17,803 种不同汉字的拆法，分为繁体字和简体字两个版本，但是其中还有大量的生僻字和古文字是没有组件标签的。

为了给这些没有组件标签的汉字生成组件标签，我们设计了一个组件分解网络，它包含基于 ResNet-18 架构的编码器和 Transformer 解码器，用于预测汉字的组件标签。输入为汉字的图片，输出为汉字的组件序列。

编码器用于提取汉字图片特征，并将其转换为一个向量（序列），送入 Transformer 解码器，以下是 Transformer 解码器的详细步骤：

- 输入嵌入：将输入的序列转换成词向量。
- 位置编码：在向量中添加位置编码，以便模型区分序列中的不同位置。
- 自注意力层：处理输入序列，使得每个位置的输出都包含序列中其他位置的信息。
- 编码器-解码器注意力层：在这个层中，解码器的每个位置都可以关注编码器的所有位置。解码器的当前位置的输出会是编码器输出的加权和，权重由当前位置和编码器输出的相似性决定。
- 前馈神经网络层：包含两个线性变换和一个非线性激活函数。
- 生成最终输出：生成每个词（组件）的概率分布，然后选择概率最高的组件作为输出。

值得注意的是，Transformer 模型的训练通常使用的是序列到序列（Seq2Seq）的学习框架，编码器和解码器是一起训练的，我们并不单独定义编码器的损失函数。损失函数主要是用来衡量解码器的输出与目标序列的匹配程度。编码器的训练是通过优化这个损失函数间接完成的。

具体来说，对于解码器在每个时间步 t 生成的输出，我们可以计算其与真实标签的交叉熵损失。如果我们的

目标序列 $\mathbf{y}$ 是一个长度为 T 的序列：$\mathbf{y} = \{y_1, y_2, ..., y_T\}$

而解码器的预测序列是：$\hat{\mathbf{y}} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_T\}$, 那么交叉熵损失可以表示为：

$$L = -\sum y_t \log(\hat{y}_t)$$

其中 y_t 是真实的标签（one-hot 编码），$\hat{y}_t$ 是模型预测的概率分布，$\sum$ 是对所有时间步 t 的求和。

组件分解网络训练完成后，我们就可以为数据集中的所有汉字生成组件标签。

接下来训练字体生成网络，我们的目标是融合参考字体的风格和源字体的内容。为此，我们在训练字体生成网络前需要先训练字体特征提取器和字体特征分类器。字体特征提取器是一个 k 头编码器，每个头称为一个专家，用于提取汉字 f_i 的局部特征，然后计算得到该组件的内容特征 $f_{c,i}$ 和风格特征 $f_{u,i}$，送入特征分类器进行分类。字体特征提取器和字体特征分类器是相辅相成的，分类器对特征提取器起到了监督的作用，让特征提取器能够提取出更有利于分类的特征。损失函数为：

$$L_{s,i}(f_{s,i}, y_s) = \mathrm{CE}\left(\mathrm{Cl}_{s,s}(f_{s,i}), y_s\right) - H\left(\mathrm{Cl}_{s,u}(f_{s,i})\right)$$

$$L_{c,i}(f_{c,i}, U_c) = L_{\mathrm{cls},c,i}(f_{c,i}, U_c) - H\left(\mathrm{Cl}_{s,s}(f_{c,i})\right)$$

其中，CE 是交叉熵损失，H 是风格独立损失。该损失函数的目的是让组件特征分类器能够成功分类组件特征，同时，不能分类风格特征，也就是让风格特征和内容特征解耦合；同理，风格特征分类器能够成功风格特征，不能分类组件特征。

字体生成网络采用了生成对抗网络（GAN）的思想，用于融合参考字体的风格特征和源字体的内容，生成目标字体。除了字体生成器外，还引入了一个分类模块 D（包含两个判别器 D_s 和 D_c 分别预测样式标签和内容标签)。引入 hinge 生成对抗损失 L_{adv}，目的是让生成器和鉴别器进行博弈，最终让鉴别器难以区分生成器生成的图片和真实的图片，从而提高生成器生成图片的质量；特征匹配损失 L_{fm}，目的是让生成器生成的图片尽可能与源字形以及目标字形的内容、风格特征匹配；像素级别重建损失 L_{recon}，目的是让生成器生成的图片拥有视觉高质量。损失函数如下，L_D 表示判别器的损失函数，L_G 表示生成器的损失函数：

$$L_D = L_{\mathrm{gdtv}}^{D}$$

$$L_G = L_{\mathrm{gdy}} + \lambda_{\mathrm{recon}} L_{\mathrm{recon}} + L_{\mathrm{fm}}$$

训练采用的部分配置如下：

- 每次训练的批次大小为 8
- 训练的最大迭代次数为 800000 次
- 生成器、判别器的学习率分别设置为 2e-4、1e-3
- 采用 Adam 优化器，beta 参数为 [0.0, 0.9]
- 对数据集进行归一化，不采用随机仿射变换
- 采用 ReLu 激活函数

训练完成后，生成器生成的字体图片与该字体生成的字体图片的相似度非常高，LPIPS 仅为 0.01。

4.3.2 基于四角编码映射的参考字数据优化模块

该模块的目标是最大化参考字多样性，具体方法为将用户所写的参考字送入百度手写汉字识别 API，识别成功后，将用户目前写好的参考字依次转换为四角码，并编码为向量，计算向量中心点，然后随机选 10 个字计算与中心点的距离，选择距离最远的汉字。

为了保证我们所推荐的汉字都是常用字，以满足用户使用体验的需求。我们在微信公众号、豆瓣论坛等 9 个经典语料库 [16] 中选取了约 10 万字的中文语料，取出其中满足上一章所提到的字形部件多样性的 2500 字，作为参考字推荐的范围。为了证明该思路的实用性，我们我们以发放问卷的形式调查了 200 位志愿者，让他们随意

写 10 个汉字，统计后发现，志愿者们所写的汉字中 99.2% 的都在该常用字列表中。结果证明本模块实际上并不会推荐一些用户不会写的汉字，同时当用户对所推荐的汉字不满意，也可以继续自由书写或者更换参考字，本模块将通过动态参考字列表更新来最大化地满足用户体验。

接下来依次介绍参考字识别、向量编码、参考字选择的设计与部署。

参考字识别本质上是一种图片到字符的转换，也就是 OCR（Optical Character Recognition，光学字符识别）技术，目前该技术已经相当成熟，并且在车牌识别、手写识别、智能阅读器等方面都已经得到了成功的应用。有许多公司和机构提供了用于 OCR 的 API 服务，这些服务通常可以直接用于识别图像中的文本。我们将本地 OCR 识别与云端 OCR 识别（调用 API）进行了对比，发现在达到相同识别准确率的情况下，本地 OCR 识别占用内存大，因此，为了降低计算开销，我们采取调用 API 的方法进行 OCR 识别。对比多个公司提供的 API 后，我们选择了百度手写文字识别 API[17]，它支持对图片中的手写中文、手写数字进行检测和识别，并针对不规则的手写字体进行了专项优化，识别准确率高达 98.4%。

接下来将用户写好的参考字转换为四角编码。为了提高转换效率以及降低存储开销，我们可以将常用汉字到其四角码的映射存储到数据库文件中。在输入汉字时，只需查询数据库即可得到四角编码，方便高效。此外，由于四角号码检字法用数字 0 到 9 表示一个汉字四角的十种笔形，我们可以依据四角编码中 0 9 出现的次数将其映射到 10 维向量空间中，用这个向量来表示汉字的结构信息：

接下来依次介绍参考字识别、向量编码、参考字选择的实验设计与部署。

参考字识别本质上是一种图片到字符的转换，也就是 OCR（Optical Character Recognition，光学字符识别）技术，目前该技术已经相当成熟，并且在车牌识别、手写识别、智能阅读器等方面都已经得到了成功的应用。有许多公司和机构提供了用于 OCR 的 API 服务，这些服务通常可以直接用于识别图像中的文本。我们将本地 OCR 识别与云端 OCR 识别（调用 API）进行了对比，发现在达到相同识别准确率的情况下，本地 OCR 识别占用内存大，因此，为了降低计算开销，我们采取调用 API 的方法进行 OCR 识别。对比多个公司提供的 API 后，我们选择了百度手写文字识别 API，它支持对图片中的手写中文、手写数字进行检测和识别，并针对不规则的手写字体进行了专项优化，识别准确率高达 98.4%。

接下来将用户写好的参考字转换为四角编码。为了提高转换效率以及降低存储开销，我们可以将常用汉字到其四角码的映射存储到数据库文件中。在输入汉字时，只需查询数据库即可得到四角编码，方便高效。此外，由于四角号码检字法用数字 0 到 9 表示一个汉字四角的十种笔形，我们可以依据四角编码中 0 9 出现的次数将其映射到 10 维向量空间中，用这个向量来表示汉字的结构信息：

$$D[i] = \text{count}(F, i)$$

映射完成后，计算这些向量的中心点，记为向量 v：

$$\mathbf{v} = \frac{\sum_{j=1}^{N} D_j}{N}$$

如果依次计算向量 v 与所有常用字之间的距离，会导致计算效率极低，因此，我们可以从常用字列表中随机选择 10 个汉字（与用户已经写过的参考字不重复），分别计算这 10 个汉字与向量 v 的距离，选择距离最大的汉字并推荐给用户，从而最大化参考字多样性：

$$P_{\text{next}} = \arg\max \left(\sum_{i=0}^{10} (P_i - C_j)^2 \right)$$

4.3.3 基于 ResNet-18 网络的源字体数据优化模块

为了提高生成图像的质量，我们部署了风格分类网络对参考字进行风格分类，然后进行特征匹配，选择与参考字风格最接近的 3 4 种字体作为源字体。

风格分类网络使用的数据集是之前介绍过的 12 种风格共 547 种字体，每种字体取出 200 张图片。数据集组成如下：

表 4.2: 字体数据集统计

风格	常用字	生僻字	古书法	字体数量	图片数量
楷书	√	√	√	55	11000
宋体	√	√	√	57	11400
行书	√	×	√	53	10600
标题字	√	×	×	28	5600
篆隶	√	×	√	36	7200
空心	√	×	×	25	5000
黑体	√	√	×	62	12400
手写体	√	×	×	67	13400
毛笔字	√	×	√	59	11800
圆体	√	√	×	57	11400
卡通体	√	×	×	26	5200
海报体	√	×	×	22	4400

因此，数据集共包含 109400 张图片，我们按照 8:2 的比例划分训练集和验证集，得到训练集图片数量为 87520，测试集图片数量为 21880。

实验超参数设置如下：

- 分类网络：ResNet-18
- 训练批次（batch_size）：128
- 损失函数：交叉熵
- 优化器：SGD，lr = 0.001，momentum=0.9，weight_decay=5e-4
- 调度器：余弦退火，T_max 为 100
- 训练轮次：100

训练完成后进行测试，分类网络在测试集上的分类准确率高达 95%。

在分类网络将参考字分类为 12 种风格的某一种（记为 i）后，我们将参考字与第 i 类风格的所有字体进行特征匹配。为了提高匹配的速率，我们事先利用字体生成模块中的多头编码器提取出每种字体的特征向量，并将其存储为 pkl 文件。每种风格的 pkl 文件数量等于该风格的字体数量，因此我们最终可以得到 547 个 pkl 文件。

进行特征匹配时，使用多头编码器提取出参考字的特征向量 v，依次计算 v 与第 i 类风格的所有字体的特征向量（假设有 ni 个）的余弦相似度，输出相似度最高的 3 4 种字体：

$$\text{top\_fonts} = \arg\max_{\{v_i\}_{i=1}^{k}} \text{Sim}(v, v_i)$$

其中，Sim 表示余弦相似度函数，k 表示我们想要的字体数量（在这个情况下，k 可以是 3 或 4）。

4.4 硬件接口设计

为了满足专业用户对字体设计的需求，本工具对数位板、手写笔等专业设计外设，基于 Wintab 协议和 HID 协议做出了适配，如下图所示：

图 4.14: 使用专业数位板进行设计

图 4.15: 使用触控屏与手写笔进行设计

第 5 章 测试分析

5.1 测试计划描述

本测试报告按照系统使用手册介绍系统的功能，测试系统的能力是否满足前文说明功能和性能需求。测试分为功能测试和性能测试两部分。功能测试覆盖各子系统中的功能模块，针对在现有项目功能模块以及实施结果分别进行测试，测试整个系统是否达到需求中要求实现的功能，以及软件运行的可靠性和测试用户界面的友好性。性能测试部分主要面向该软件的三个模块，即核心生成模型、参考字数据优化模块、源字体数据优化模块分别从字体生成效果和字体生成时间等方面设计了对比实验和消融实验，以测试软件的实用性和高效性。

5.2 功能测试

5.2.1 基本功能测试

表 5.1: 基本功能测试

序号	测试用例	预期结果	实测结果	测试状态	错误类型
测试单元：参考字体绘制					
1	左侧工具栏	可流畅调整笔画的粗细、画布的缩放以及相对位移；可自定义压感曲线；标尺和橡皮擦功能正常	与预期结果一致	1	
2	上方笔触栏	可选择并切换包括签字笔、钢笔、毛笔、荧光笔等笔触效果	与预期结果一致	1	
3	画布	可流畅绘制参考字型，支持手写笔或压感屏的压感效果。	与预期结果一致	1	
4	下方参考字选择	可自由选择已经写入的参考字，并且支持修改	与预期结果一致	1	
测试单元：下一字推荐					
5	参考字识别	根据用户绘制的图像自动进行汉字 OCR 以得到参考字列表	与预期结果一致		
6	右下方 AI 助手	可根据当前已经提供的参考字基于字形多样性最大化原则提供下一个参考字推荐	与预期结果一致	1	
7	参考字动态调整	当参考字有变动时，检测出更改，并动态更新列表	与预期结果一致	1	
测试单元：源字体匹配					
8	源字体匹配	对当前的参考字字形进行风格抽取并在字库之中选择风格最相似的前 4 个字体作为候选源字体	与预期结果一致	1	
9	中间栏源字体展示	在中间展示出所匹配的四个最为相似的字体，可供用户选择	与预期结果一致	1	
测试单元：字体生成					
10	风格迁移模型	根据提供的参考字和所选择的源字体，将参考字的风格迁移至源字体的字形中，生成目标字体	与预期结果一致	1	
11	效果预览	用户可自定义 8 个字以内的预览句子，在右上方的预览格子内呈现所生成的目标字体的效果	与预期结果一致	1	

5.2.2 可靠性测试

表 5.2: 可靠性测试

测试用例	测试过程描述	测试结果
成熟性	输入参考字容量达到规定的极限时，系统不崩溃、不异常退出也不丢失数据	系统在达到极限时提示参考字足够
	生成模型占用内存/显存超出可分配极限时，系统不崩溃，不异常退出也不丢失数据	系统在内存/显存不足时向用户提醒，并等待
	项目描述中列出的其他程序或用户造成的错误输入时，系统不崩溃也不丢失数据	存在错误录入时系统给出相应的提示信息
	输入用户文档中明确规定的非法指令时，系统不崩溃也不丢失数据	输入用户文档中明确规定的非法指令时，系统给出相应的提示；如上传文件格式不符合上传允许的格式规范
容错性	能屏蔽用户的误操作	系统符合该项操作，如输入的预览字超过 8 个后，默认选取前 8 个展示生成效果
	对错误有正确提示	系统给出相应的提示信息
	输入错误数据时，系统不崩溃、不异常退出也不丢失数据	系统给出相应的提示信息
	有错误操作时，系统不崩溃、不异常退出也不丢失数据	系统给出相应的提示信息
易恢复性	系统运行失效后可以较快重建系统	系统符合该要求
数据校验机制	应对模型数据之间的逻辑关系进行校验，保证数据的有效性	系统符合该项操作，如验证生成模型的批次大小，图片的像素大小等
	应保证数据的完整性和一致性，不会因删除或反复的更新而被破坏或留下垃圾数据	系统更新或删除功能不影响系统数据
	对不符合要求的输入数据，系统应使用中文给出简洁、准确的提示信息，必要时应给出帮助	系统对于不符合要求的输入给出相应的提示信息

5.2.3 易用性测试

表 5.3: 易用性测试

测试用例	测试过程描述	测试结果
页面风格一致性	页面结构、导航、菜单、文本框、翻页、字体、列表、日期和绘画板控件、数据精度的风格是否一致	系统页面风格一致
易浏览性	具有必要的信息，指导用户使用程序	系统界面组件提示操作信息，方便指导用户操作
	输入、输出设计规矩，输出结果应简洁、直观、美观、方便阅读、易懂和使用	系统界面显示简洁易懂，方便使用
	人机界面简洁、美观、实用，风格相对一致，符合办公习惯	系统操作简单易用
	在界面、人机交互、输出中的用语应与业务用语一致	系统操作简单易用

5.3 性能测试

性能测试着眼于本项目提出的几个算法和模块，以及所构建的数据集，我们一方面设计了对比和消融实验，通过多种不同的评价指标来证明我们所设计的优化模块和构建的数据集对模型优化起到正面效果。以充分说明本项目提出方法的创新性和实用性。另一方面横向比较了各种方法在字体生成时的时间/人力成本开销，以充分说明本项目提出方法的高效率与可部署性。

5.3.1 评价指标介绍

在实验中，为了衡量模型生成的字体效果，我们使用了包括 LPIPS[2](学习感知图像补丁相似度)、MSE(均方误差)、PSNR[10](峰值信噪比)，以及真人辨识率作为评价指标，四种指标的介绍如下：

表 5.4: 性能测试评价指标

评价指标	原理	评价方法
LPIPS	学习感知图像补丁相似度，用于衡量两张图片在人类视觉感知上的差异的指标。LPIPS 越小，表示两张图片在人类视觉上的差异越小，即它们越相似；反之，LPIPS 越大，表示两张图片在人类视觉上的差异越大，即它们越不相似	使用参考字型与生成目标字形图片计算 LPIPS，**值越小**说明生成字形的风格还原度越高
MSE	均方误差，两个字体图片的相似度可以通过计算它们像素值之间的均方误差来实现。均方误差越小，表示两个图片的相似度越高。对于像素和分辨率不一样的图片，需要经过裁剪	使用参考字型与生成目标字形图片计算 MSE，**值越小**说明生成字形的风格还原度越高
PSNR	峰值信噪比，峰值信噪比越高，表示两个图片的相似度越高	使用参考字型与生成目标字形图片计算 PSNR，**值越大**说明生成字形的风格还原度越高
真人辨识率	我们随机选择了 200 位志愿者，分别向其展示生成的字体与参考字体，并令其尝试分辨，分辨成功的概率可以反应两种字体在人眼看来的区别度	真人辨识率越低（接近 50%），说明生成字形的风格还原度越高，生成效果越好

5.3.2 效果测试一：核心生成模型生成效果测试

本节测试了团队所提出的基于多头编码器的字体生成模型，与基于开源代码的批量生成方案和目前成熟的 Cycle GAN（以下简称 GAN）和 FUNIT 这两种 AI 风格迁移模型。四种方法分别在楷体、圆体、黑体这三种风格的字体生成任务中进行了对比实验，实验结果和可视化展示如下表和下图所示：

表 5.5: 核心生成模型生成效果对比测试结果

方法	楷体				圆体				黑体			
	LPIPS↓	MSE↓	PSNR↑	Dis_h↓	LPIPS↓	MSE↓	PSNR↑	Dis_h↓	LPIPS↓	MSE↓	PSNR↑	Dis_h↓
开源字体修改	0.334	17.97	31.26	0.899	0.339	17.88	31.21	0.945	0.309	17.25	31.56	0.964
GAN	0.236	14.78	33.53	0.654	0.232	14.63	33.60	0.649	0.220	14.61	33.70	0.620
FUNIT	0.210	13.66	34.82	0.608	0.201	13.58	34.88	0.610	0.198	13.45	34.99	0.591
多头编码器	**0.112**	**11.27**	**37.39**	**0.545**	**0.116**	**11.20**	**37.43**	**0.569**	**0.104**	**11.12**	**37.62**	**0.523**

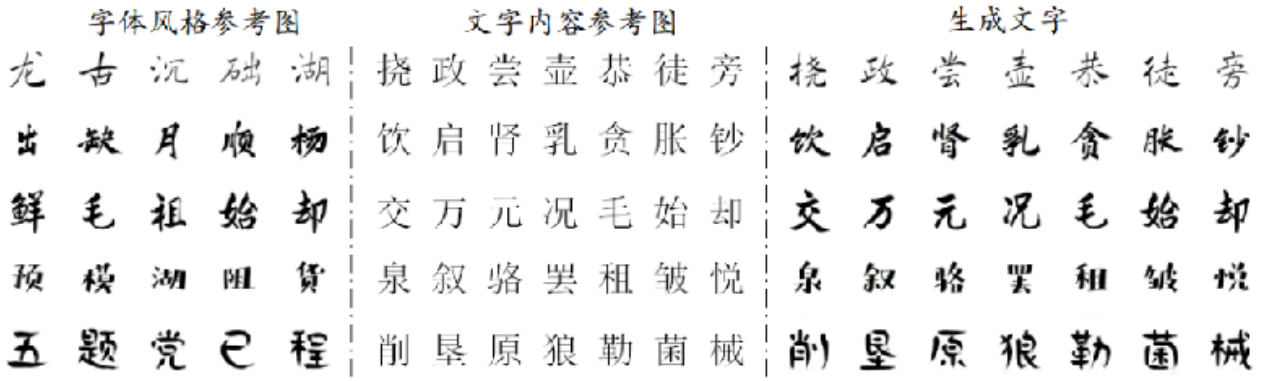


图 5.1: 核心生成模型生成效果对比测试可视化展示

如上图/表所示，基于开源字体修改的方法，因为所选择的开源字体（楷体、幼圆、微软雅黑）与目标字体的区别较大，所以无论怎么修改都无法达到目标风格的要求。而传统的生成网络在如“算”、“赛”等结构较为复杂的汉字时，生成效果不好，本项目提出的基于多头编码器的网络则解决了这一问题，在各种风格的字形生成任务上都获得了优良的表现，相比起传统 AI 生成模型，LPIPS 和真人辨识率上的平均表现分别获得了 33.94% 和 26.79% 的提升。

5.3.3 效果测试二：数据优化模块对生成效果影响的消融实验

本节测试了团队所提出的两个数据优化模块，设计了消融实验，分别从参考字数据和源字体数据的角度验证所提出的两个基于大数据的优化模块对生成效果起到促进作用。实验效果和可视化展示如下表和下图所示：

表 5.6: 两个数据优化模块为生成效果带来的提升

方法	楷体				圆体				黑体			
	LPIPS↓	MSE↓	PSNR↑	Dis_h↓	LPIPS↓	MSE↓	PSNR↑	Dis_h↓	LPIPS↓	MSE↓	PSNR↑	Dis_h↓
G	0.183	12.89	35.26	0.584	0.186	12.96	35.23	0.587	0.170	12.76	35.35	0.577
G, M_{ref}	0.156	12.00	36.14	0.554	0.157	12.06	36.18	0.549	0.148	12.01	36.25	0.541
G, M_{source}	0.127	11.39	37.01	0.519	0.132	11.42	37.06	0.523	0.125	11.37	37.38	0.519
G, M_{source}, M_{ref}	**0.112**	**11.27**	**37.39**	**0.505**	**0.116**	**11.20**	**37.43**	**0.509**	**0.104**	**11.12**	**37.62**	**0.503**

图 5.2: 两个数据优化模块为生成效果带来的提升-可视化展示

如上图/表所示，除了修改模型结构，对输入模型的数据进行优化对生成效果的提升也有很大帮助，本团队设计了基于四角编码映射的参考字数据优化模型和基于 ResNet-18 的源字体数据优化模型，借助现代字体库和生僻字/古书法数据集，解决了目前 AI 字体生成领域面临的部件多样性与风格杂糅两大挑战。使用这两个数据

优化模块，相比起直接送入字形数据，在 LPIPS 和真人辨识率这两个指标上的平均表现获得了 8.45% 和 12.01% 的提升。

5.3.4 效果测试三：大数据对模型生成效果影响的消融实验

本节对团队所提出的包含了现代字体库 + 生僻字库 + 古书法字库三大数据集优化模型训练的思想对最终模型生成的效果的影响进行验证。我们分别使用规模为 1000 50000 字的现代字体库，以及使用对生僻字/古书法字体集的情况进行了消融实验，实验结果如下图所示：

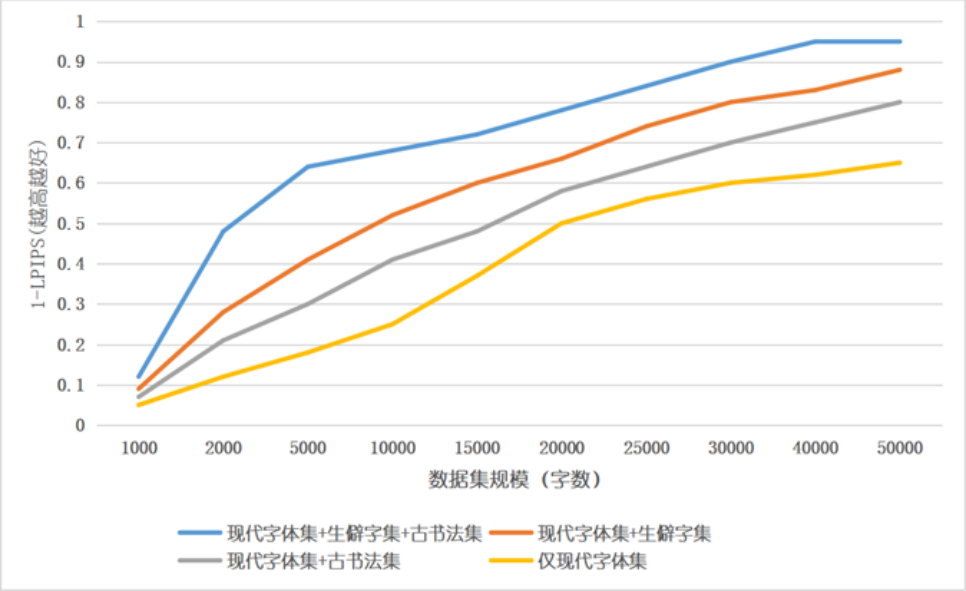


图 5.3: 大数据（数据集规模/数据集多样性）对模型 LPIPS 影响的消融实验结果

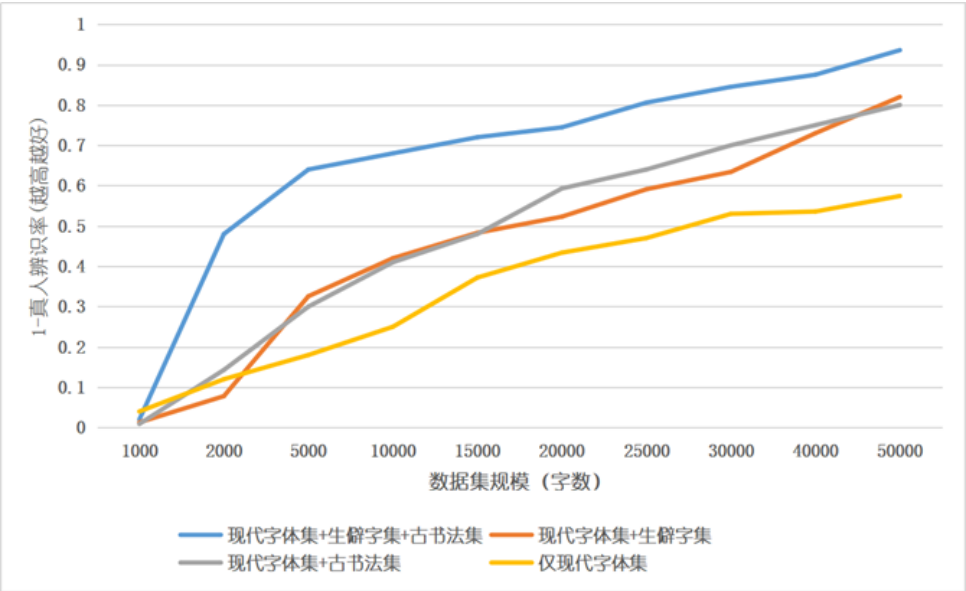


图 5.4: 大数据（数据集规模/数据集多样性）对模型真人辨识率影响的消融实验结果

实验结果表明，数据集规模的扩大和使用多种不同类型的数据集，有利于提高模型的生成效果。印证了我们团队使用生僻字库和古文字来优化复杂/非经典字形生成表现的猜想。也充分证明了大数据思想在优化 AI 模型上起到的正面作用。

5.3.5 成本测试一：AI 生成与传统方法消耗成本对比测试

为了证明使用本工具利用 AI 生成方法能很好地解决传统逐字设计方法所面临的流程繁琐成本高的痛点，我们从人力/时间成本消耗的角度进行了对比测试：我们分别选择了一位专业设计师使用本工具/进行逐字设计和 5 位专业设计师共同进行逐字设计，对以上三种实验条件进行了对比实验，结果如下表所示：

表 5.7: AI 生成与传统方法消耗成本对比实验

	笔韵智枢	逐字绘制	
LPIPS	0.127	0.116	0.124
Dis_h	0.519	0.506	0.515
时间	10min	24day	5day
设计师数量	1	1	5

从上表可以看到，一个设计师使用 4 分钟，就可以完成一个团队 5 天才能完成的工作（达到类似的 LPIPS 等指标），本工具极大地缩短了字体的制作周期和人力成本，解决了长久以来困扰字体设计师和字体设计行业的痛点。

5.3.6 成本测试二：参考字数据优化模块对成本开销影响的消融实验

为了证明所设计的源字体数据优化模块，能够在优化模型生成效果的同时，减少所需要提供的参考字数，以提高软件使用的效率，实现“大数据助力小样本”的构想。

表 5.8: 源字体数据优化模块对成本开销影响的消融实验结果

	使用数据优化	无优化
LPIPS	0.127	0.183
Dis_h	0.519	0.513
所需参考字数	10	200-300
耗费时间	3min	25min

从上表可以看到，对源字体进行数据优化能将原有的数百个参考字缩减为 10 个，大大降低了参考字绘制过程的繁琐程度，实现真正意义上的少样本字体生成，优化了用户体验。

5.3.7 成本测试三：源字体数据优化模块对成本开销影响的消融实验

与上节相同，设计消融实验，实验结果如下表：

表 5.9: 源字体数据优化模块对成本开销影响的消融实验结果

	使用数据优化	无优化
LPIPS	0.130	0.116
Dis_h	0.533	0.506
迭代次数	1	10
耗费时间	<5s	6min

上表的实验结果说明了，对源字体进行的数据优化旨在通过对所提供的包括现代、生僻字、古书法三大数据集的学习中，学会对字体风格进行理解的能力，为参考字匹配最相似的源字体，最终省去迭代寻找源字体这一流程。

5.3.8 成本测试四：工具在不同硬件设备上的运行成本测试

为了验证本工具在不同的硬件环境下都有充分的可部署性，我们在不同硬件配置的主机上，对参考字推荐、源字体匹配、字体生成三个主要流程的耗时进行了测试，测试结果如下表所示：

表 5.10: 本工具在不同配置的硬件环境下的时间开销

实验主机配置		平均耗时（秒）		
GPU	**CPU**	**参考字推荐**	**源字体匹配**	**字体生成**
RTX3090	Intel Platinum 8255C	0.093	0.566	23.635
RTX2080	Intel Platinum 8255C	0.098	0.978	56.317
GTX 1650	Intel Core i7-10750H	0.103	2.611	117.679
无	AMD R7 5825U	0.084	6.313	289.74

可以看到，即使是在无 GPU 的较低配置的主机上，本工具也能在数分钟内生成所需的字体。以上实验证明本工具对硬件配置的并没有太高的要求，可以广泛部署应用。

第 6 章 作品总结

6.1 作品特色与创新点

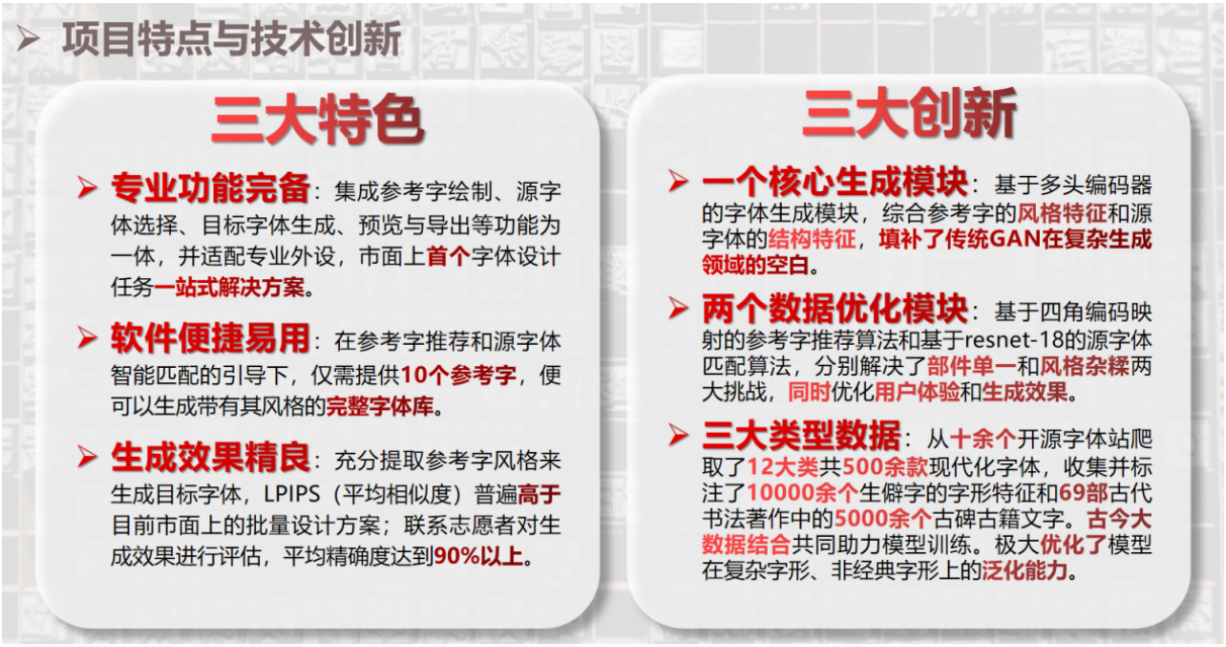


图 6.1: 三大特色、三大创新

6.1.1 三大特色

本作品主要针对字体设计任务目前所面临的流程繁琐成本高、风格局限效果差等痛点，采取 AI 字体生成的思路辅以现代 + 生僻字 + 古书法三大字库数据集，大数据助力少样本模型优化，并且设计了方便易用的软件界面。以代替字体大批量逐字设计的繁琐工序。本作品的应用特色可概括为软件功能完备、使用方便快捷、生成效果精良三点。

- **软件功能完备：**工具集成参考字绘制、源字体选择、目标字体生成、预览与导出等功能为一体，为字体设计任务提出并实现了一站式的解决方案。其内置的画布满足了专业绘图所需的自定义压感、笔锋笔触，以及选取等标尺等功能。同时我们基于 Wintab 协议和 HID 协议对数位板、手写笔等专业外设做出了适配，可以满足严肃字体设计的需要。
- **使用方便快捷：**本工具部署了基于生成对抗网络的 AI 字体生成模型，辅以大数据作为支撑的源字体和参考字数据优化模块，同时实现了生成效果和用户体验的优化。本项目在实地部署过程中的应用结果表明，**一位字体设计师**使用本工具进行字体创作，可以在 **10 分钟内**完成原先需要 **5 位专业设计师数十天**才能完成的设计工作。
- **生成效果精良：**在传统的生成式神经网络模型的基础上，我们设计了**多头编码器**分别抽取字形中各部件的风格/结构特征，并通过大数据技术助力，构建了**现代字体库 + 生僻字体库 + 古书法库三大数据集**，以优化模型表现。在包括但不限于楷书、宋体、标题字、手写体、行书等不同风格字体的生成效果测试中，本工具的平均相似度（LPIPS）指标优于市面上其它模型的表现，同时我们邀请了 200 余位志愿者通过辨认生成字体与理想目标生成效果的方式来评估本工具的生成效果，测试结果表明本工具所生成的字体效果有超过 90% 的准确率。

6.1.2 三大创新

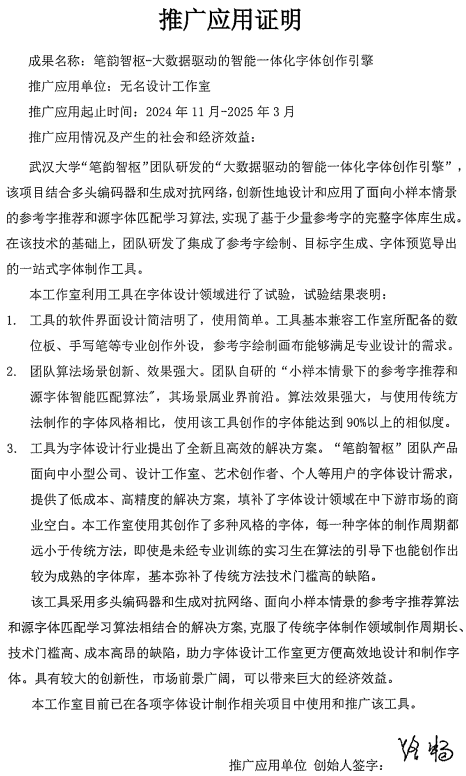
本团队独创性地提出了 AI 生成字体的方法，应用于字体设计行业之中，并针对 AI 字体生成模型面临的参考字部件多样化、风格杂糅、复杂字形生成效果差的三大技术挑战，提出了**“一个核心模型 +2 个数据优化模块 +3 大类型数据集”**的解决方案，旨在**以大数据助力小样本生成**，优化模型应用表现。

- **一个核心生成模块：**针对字体生成这一图像生成领域的垂类，本团队设计了专用于多头编码器的字体生成模型，并辅以匈牙利算法，实现字体部件风格/结构特征的自动抽取和匹配，填补了传统 GAN 在复杂生成领域的空白。
- **两个数据优化模块：**针对参考字部件多样化和风格杂糅两大技术挑战，本团队从数据的角度切入，分别设计了基于四角编码映射的**参考字推荐算法**和基于 ResNet-18 深度网络的**源字体匹配算法**，使用三大数据集进行辅助构建，同时优化了用户体验和生成效果。
- **三大类型数据：**针对传统的少样本生成网络在复杂字形和非经典字形上表现不好的问题，本团队独创性地引入**大数据思想**，从十余个开源字体站爬取了 12 大类共 500 余款**现代化字体**，收集并标注了 10000 余个**生僻字**的字形特征和 69 部古代书法著作中的 5000 余个**古碑古籍文字**。**古今三大数据集结合共同助力模型训练**。极大优化了模型在复杂字形、非经典字形上的泛化能力。也使得本工具能很好地胜任字体在生僻字域上的扩展和古书法修复的任务，兼具经济和文化效益。

6.2 应用推广

6.2.1 实地部署——赋能小企业小工作室字体创作

目前，本项目已经进入应用测试阶段。我们与武汉无名设计工作室、武汉未知比特信息科技有限公司分别达成了合作。在字体设计领域开展了实际应用，帮助平面设计师设计并制作其个性化风格的字体，并在产品设计、广告制作、视频编辑等工作中提供支持。项目得到了实地应用单位的一致好评。



推广应用证明

成果名称：笔韵智枢-大数据驱动的智能一体化字体创作引擎

推广应用单位：无名设计工作室

推广应用起止时间：2024 年 11 月-2025 年 3 月

推广应用情况及产生的社会和经济效益：

武汉大学“笔韵智枢”团队研发的“大数据驱动的智能一体化字体创作引擎”，该项目结合多头编码器和生成对抗网络，创新性地设计和应用了面向小样本情景的参考字推荐和源字体匹配学习算法，实现了基于少量参考字的完整字体库生成。在该技术的基础上，团队研发了集成了参考字绘制、目标字生成、字体预览导出的一站式字体制作工具。

本工作室利用工具在字体设计领域进行了试验，试验结果表明：

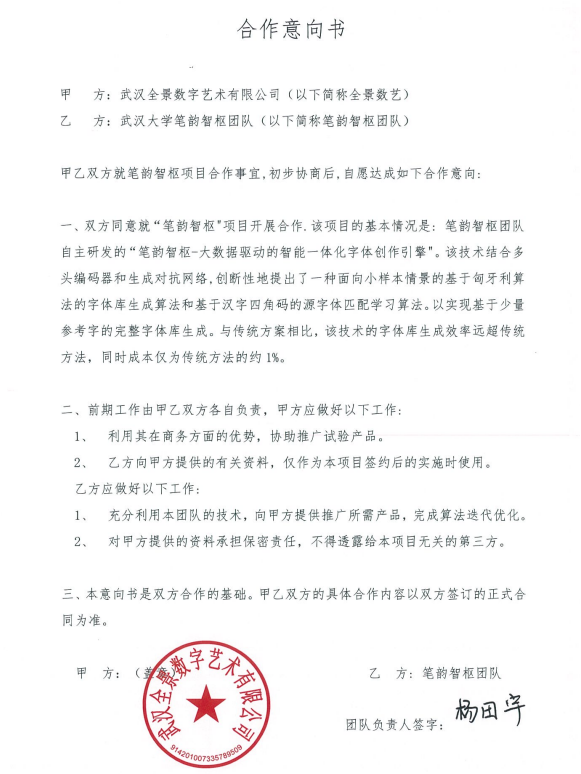
1. 工具的软件界面设计简洁明了，使用简单。工具基本兼容工作室所配备的数位板、手写笔等专业创作外设，参考字绘制画布能够满足专业设计的需求。
2. 团队算法场景创新、效果强大。团队自研的“小样本情景下的参考字推荐和源字体智能匹配算法”，其场景属业界前沿。算法效果强大，与使用传统方法制作的字体风格相比，使用该工具创作的字体能达到 90%以上的相似度。
3. 工具为字体设计行业提出了全新且高效的解决方案。“笔韵智枢”团队产品面向中小型公司、设计工作室、艺术创作者、个人等用户的字体设计需求，提供了低成本、高精度的解决方案，填补了字体设计领域在中下游市场的商业空白。本工作室使用其创作了多种风格的字体，每一种字体的制作周期都远小于传统方法，即使是未经专业训练的实习生在算法的引导下也能创作出较为成熟的字体库，基本弥补了传统方法技术门槛高的缺陷。

该工具采用多头编码器和生成对抗网络、面向小样本情景的参考字推荐算法和源字体匹配学习算法相结合的解决方案，克服了传统字体制作领域制作周期长、技术门槛高、成本高昂的缺陷，助力字体设计工作室更方便高效地设计和制作字体。具有较大的创新性，市场前景广阔，可以带来巨大的经济效益。

本工作室目前已在各项字体设计制作相关项目中使用和推广该工具。

推广应用单位 创始人签字：

图 6.2: 本作品在某工作室开展实地推广应用



合作意向书

甲　方：武汉全景数字艺术有限公司（以下简称全景数艺）
乙　方：武汉大学笔韵智枢团队（以下简称笔韵智枢团队）

甲乙双方就笔韵智枢项目合作事宜，初步协商后，自愿达成如下合作意向：

一、双方同意就“笔韵智枢”项目开展合作。该项目的基本情况是：笔韵智枢团队自主研发的“笔韵智枢-大数据驱动的智能一体化字体创作引擎”。该技术结合多头编码器和生成对抗网络，创新性地提出了一种面向小样本情景的基于匈牙利算法的字体库生成算法和基于汉字四角码的源字体匹配学习算法。以实现基于少量参考字的完整字体库生成。与传统方案相比，该技术的字体库生成效率远超传统方法，同时成本仅为传统方法的约 1%。

二、前期工作由甲乙双方各自负责，甲方应做好以下工作：
1、　利用其在商务方面的优势，协助推广试验产品。
2、　乙方向甲方提供的有关资料，仅作为本项目签约后的实施时使用。
乙方应做好以下工作：
1、　充分利用本团队的技术，向甲方提供推广所需产品，完成算法迭代优化。
2、　对甲方提供的资料承担保密责任，不得透露给本项目无关的第三方。

三、本意向书是双方合作的基础。甲乙双方的具体合作内容以双方签订的正式合同为准。

甲　方：（盖章）　　乙　方：笔韵智枢团队

团队负责人签字：杨田宇

图 6.3: 本作品在某信息科技有限公司开展实地推广应用

6.2.2 落地合作——腾讯输入法“汉字守护计划”

除此之外。本团队已经与腾讯公司接触，并达成初步合作意向。并利用“笔韵智枢”工具的生僻字与扩展功能，积极参与腾讯输入法与工信部电子标准院共同开展的“汉字守护计划”。

图 6.4: 腾讯输入法“汉字守护计划”

最后，团队将继续在学校的大力支持下，积极寻求其它设计工作室和有字体设计需求的个人/公司展开进一步合作，争取早日全面迭代项目功能，投入实际使用。助力更多的个人和团队进入字体设计领域，为我国汉字文化的发展注入新的活力。

6.3 作品展望

6.3.1 应用前景

本项目有着广阔的应用前景和良好的社会影响，具体分析如下：

- 促进完善汉语字库：我国目前字体种类和风格多样性仍有较大空缺，本项目大大降低了字体创作的门槛，任何团队和个人，借助笔韵智枢工具，都能方便而轻松地创作属于自己的字体，从而为汉语字库注入新鲜血液，使我国字库种类和风格多样性得到进一步完善。
- 促进书法文化的保护与发展：书法是我国传统文化的重要组成部分，从古至今华夏大地上诞生了无数的书法作品和书法风格，但随着时间的推移，其中的很大一部分作品和其风格都被淹没在时间长河之中。借助笔韵智枢工具，可以从古籍、碑文里残存的文字之中提取出风格，并还原出完整的字体数据。让书法文化在数字时代得到更好的保护。
- 助力生僻字字库的建立：目前有史料可查的汉字已经超过了 30 万字，但是我们的常用字符集大多还停留在不到 1 万字的层面。借助笔韵智枢工具，可以将当前的字体作为参考字用于字体生成，以将其字体风格扩展到完整汉字域上。
- 助力小企业/小工作室字体创作：目前字体制作技术门槛高、人力物力消耗大，大型公司对行业的垄断严重，而对于在广告制作、产品设计等方面有着创意字体应用需求的小型企业/工作室，囿于版权问题，很难找到合适且可商用的字体。借助笔韵智枢工具，花费较短的时间和较少的人力物力，便可以按照需求创造出合适的字体以投入实际应用。

6.3.2 后续升级方向

到目前为止，本项目已经初步完成并投入实地应用，但它也并非十全十美，也有需要改进的地方和未来需要进一步完善、扩展的方向。具体来说后续升级方向规划如下：

- 技术栈迁移：前期开发时，使用了部分较为落后的技术栈，如客户端的 pyGame 库和 tkinter 库，计划在年底前将落后的技术栈完全移除，并完全迁移到 QT 框架作为客户端的主要技术栈。
- 多平台扩展：本工具目前只支持 windows 平台客户端，实际上目前有一部分设计工作室正在采用 Android 或是 iOS 平台作为主要生产力平台（例如 iPad+Apple pencil 的解决方案），我们计划在 7 月底之前开发出对应系统的客户端并对对应的硬件外设做出适配。
- 新功能拓展，未来，团队还计划将软件设计成一款满足更专业要求的字体设计工具，添加团队协作功能，使得一个团队中的多个设计者能对打开的同一个工程进行实时共同编辑和修改（实际上在本软件当前版本的 UI 和底层架构、中已经初步预留了该功能的接口），但碍于目前团队成员技术技能限制，尚未完全掌握其中所需要的高速网络编程等技术，计划在今年年底前学习对应技术，并尽早对项目进行进一步的更新。

除了技术上的升级，团队还将与更多企业、设计工作室、以及有应用需求的个人进行交流合作与应用部署，切实为祖国的字库产业发展尽一份绵薄之力。

参考文献

[1] 李克强. 李克强在第十三届全国人大一次会议上的政府工作报告（全文）. http://scio.gov.cn.

[2] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 586–595.

[3] 字由. 字由-让用字更自由. http://hellofont.cn.

[4] 100font.com. *100font.com - 免费字体下载 - 免费商用字体下载网站*. http://100font.com.

[5] 求字体网. 字体下载-求字体网提供中文和英文字体库下载、识别与预览服务，找字体的好帮手. http://qiuziti.com.

[6] 字魂网. 字体下载 — —. http://izihun.com.

[7] 国家图书馆. 读者云门户. 中华古籍资源库. http://nlc.cn.

[8] fighting41love. *fighting41love/funNLP*.

[9] 百度. *https://console.bce.baidu.com/ai/*. https://console.bce.baidu.com/ai/.

[10] James A. Moorer. *The Use of the Phase Vocoder in Computer Music Applications*.

[11] 中华人民共和国教育部. 新版《信息技术中文编码字符集》强制性国家标准发布. http://moe.gov.cn.

[12] DrXie. *DrXie/OSFCC: 中文开源字体集*. http://github.com.

[13] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2223–2232.

[14] Tero Karras, Samuli Laine, and Timo Aila. “A Style-Based Generator Architecture for Generative Adversarial Networks”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4401–4410.

[15] Seonghyeon Park et al. “Multiple Heads Are Better Than One: Few-Shot Font Generation with Multiple Localized Experts”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 13900–13909.

[16] Harold W. Kuhn. “The Hungarian Method for the Assignment Problem”. In: *Naval Research Logistics Quarterly* 2.1-2 (1955), pp. 83–97.

[17] Wikipedia. 四角号码 - 维基百科，自由的百科全书. http://wikipedia.org. 2020.

[18] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.