成　绩：_____

汕头大学本科生课程期末项目报告

人机交互·深度学习&手绘

Visionary Sketches:

From Digit Recognition to Image Retrieval

院（系）名 称： 数学与计算机学院

专 业 名 称： 计算机科学与技术

成 员 ： 林泓宇 2022631024

汕头大学

2024 年 01 月 15 日

一、项目目标

本项目旨在通过一系列综合性任务，帮助我们全面掌握人机交互设计、深度学习模型应用、以及前端与后端技术的结合应用。项目采用 MVC 架构进行开发，要求我们在实现过程中充分考虑系统的模块化、可维护性和扩展性，以确保代码质量和项目的可持续发展性。具体目标如下：

· **掌握人机交互设计的基本概念与页面设计原则**

本项目的基础是帮助我们理解人机交互（Human-Computer Interaction，HCI）领域的核心概念，并将这些理论应用于实际的界面设计中。我们将学习 UI 设计的美学、可用性和交互设计中的用户体验（UX）优化，并在此基础上设计具有流畅交互和视觉吸引力的画板界面。该界面应能够支持基本的手写数字输入和图像交互，并且设计风格应符合现代前端开发标准，同时注重响应性和可访问性。

· **前端画板界面设计与开发**

本项目将使用前端开发技术（如 HTML5、CSS3、JavaScript、jQuery 等）设计和实现一个画板界面，满足人机交互设计的基本原则，具备手写数字识别和手绘图像交互功能。画板核心功能包括但不限于数字手写输入、图像清除、图像保存等，我们将注重前端页面的交互性、用户友好性和响应式设计，确保系统的界面美观且功能齐全。

· **基于 MVC 架构的深度学习手写数字识别项目开发**

本项目要求我们基于 Flask 框架，采用 MVC（Model-View-Controller）架构原则，开发一个手写数字识别系统。本项目将设计并训练一个深度学习模型，使用全连接神经网络（FCN）和卷积神经网络（CNN）两种架构对手写数字进行识别。通过实现和优化这些模型，我们将分析两者在识别效果、运行效率、模型参数量以及网络层数等方面的差异，并根据这些分析结果优化模型性能。我们需要确保项目符合 MVC 架构的设计原则，合理分离模型层、视图层和控制层，保证代码结构清晰、易于维护和扩展。

· **深度学习模型设计与性能分析**

本项目将在手写数字识别项目中设计并训练深度学习模型，并将其嵌入到前端设计的画板界面中，实现交互式识别。我们将分析全连接神经网络与卷积神经网络在不同方面（如运行效率、参数量、层数等）对项目性能的影响，并通过表格形式进行对比分析。为了帮助更好地理解模型架构，我们将提供详细的网络结构图，并对每一层进行文字描述，阐述每一层的功能及其对整体模型性能的影响。

· **手绘图像检索代码复现**

本项目将根据提供的开源代码库（SketchyDatabase）复现手绘图像检索功能。通过代码复现，我们将深入理解深度学习模型在图像检索中的应用，并掌握如何将深度学习技术与图像检索系统相结合。此部分任务要求我们提供详细的网络结构图和核心代码，帮助我们加深对图像检索算法及其实现细节的理解。

· **项目实验结果的可视化展示与视频录制**

本项目将对手写数字识别和手绘图像检索的实验结果进行合理的可视化展示，确保展示的实验数据清晰且易于理解。展示内容包括但不限于模型的训练结果、识别效果的准确性、不同方法的性能对比等。我们将使用可视化工具（如 matplotlib、Plotly 等）展示数据，并通过合理的图表进行呈现。此外，我们将制作项目展示视频，视频需展示人机交互界面的操作过程，并在不超过 20MB 的文件大小限制下提交，确保实验结果的完整性与可视化效果。

二、项目软硬件设备情况

· 主机：Legion R9000P 2022

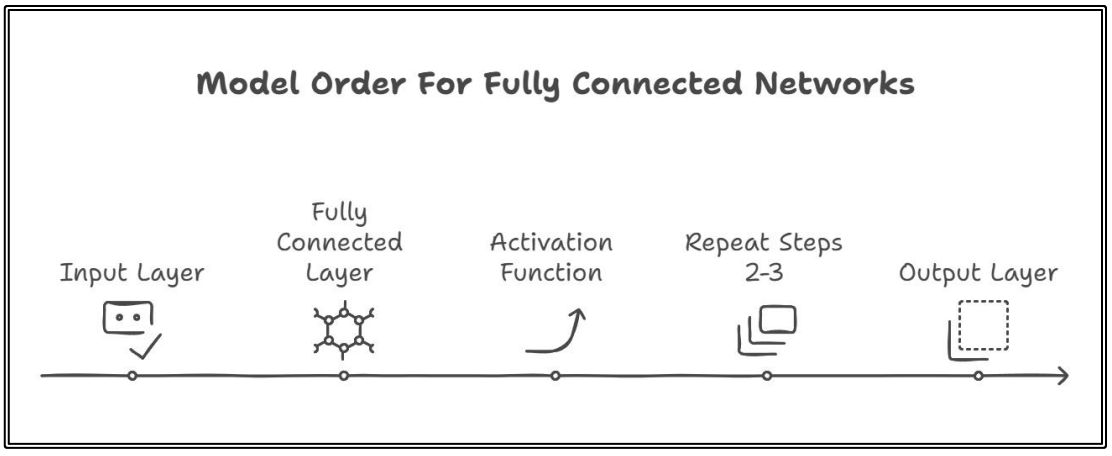
· 操作系统：Ubuntu 24.10

· CPU：AMD

· GPU：NVIDIA GeForce RTX3060（实验选用 CUDA 核心进行训练）

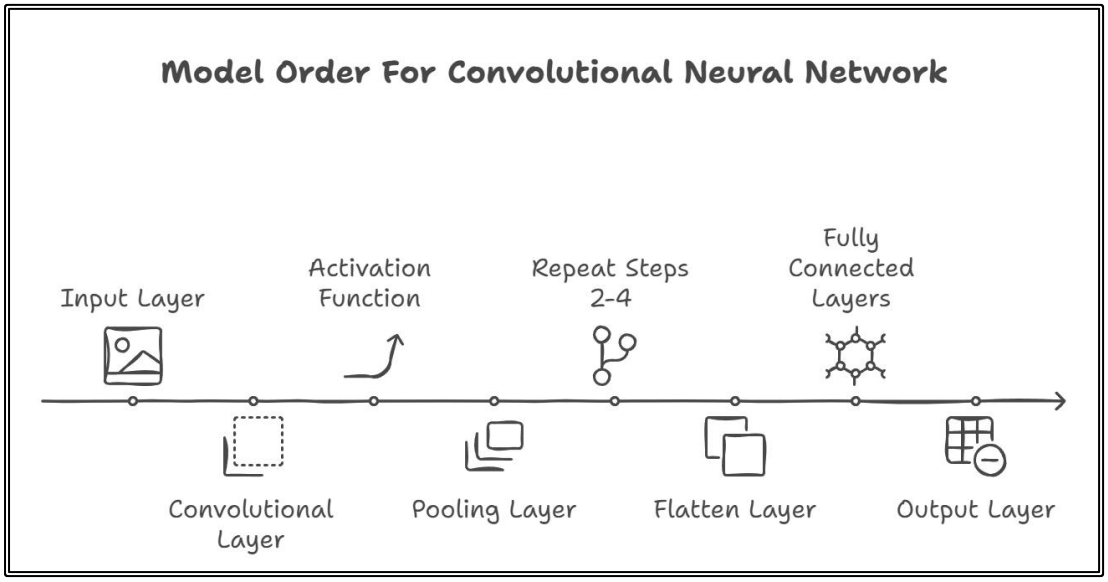
三、项目模型概述 · 全连接神经网络 FCN 与卷积神经网络 CNN

1）全连接神经网络的模型架构——FCN



图一 全连接神经网络(FCN)模型架构

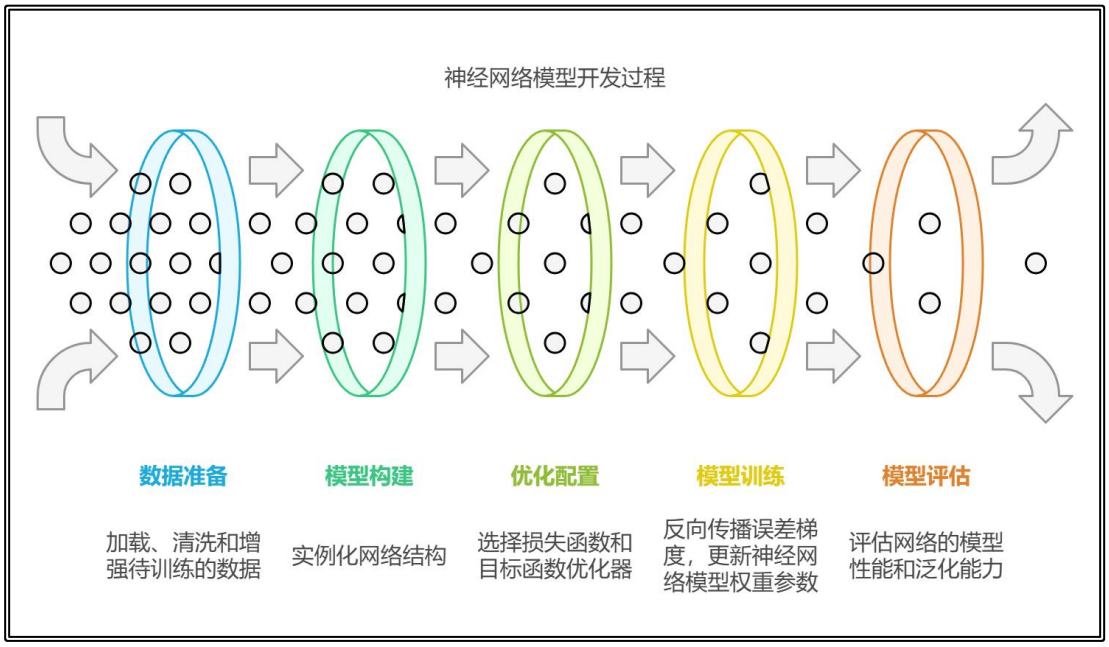
2）卷积神经网络的模型架构——CNN



图二 卷积神经网络(CNN)模型架构

四、项目实现过程一·全连接神经网络与卷积神经网络的训练过程

Part1：神经网络通用训练流程

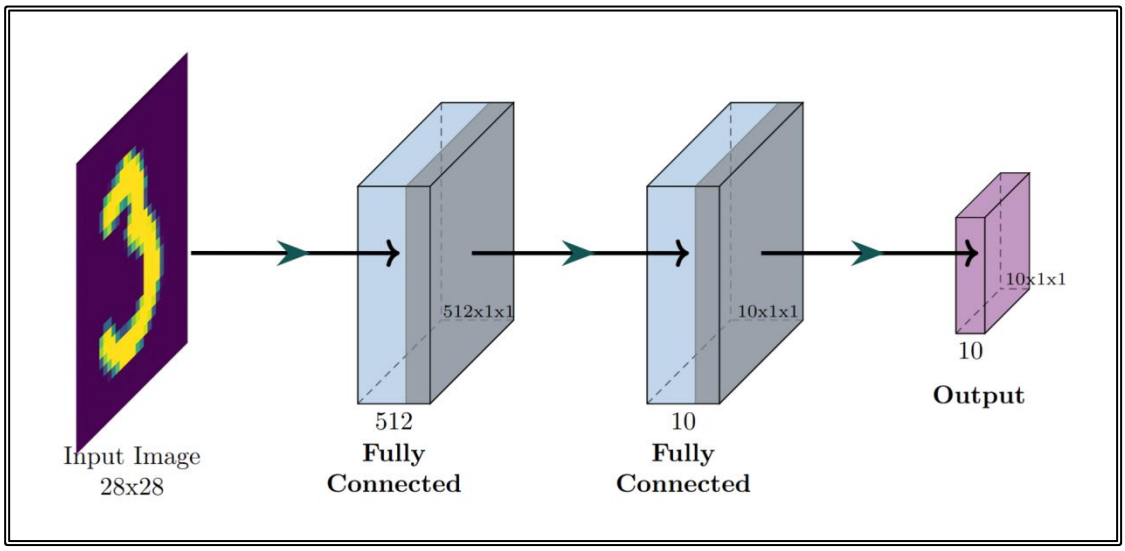


图三 神经网络之通用训练流程

Part2:全连接神经网络 FCN 与卷积神经网络 CNN 模型架构图

1）FCN 全连接神经网络之模型架构图及相应阐释

该全连接网络模型，旨在实现对 MNIST 数据集的图像分类任务。模型首先接收 28x28 像素的单通道图像，并将其展平为 784 维的向量，作为网络的初始输入。随后，此向量被送入第一个全连接层（fully connected layer1），该层通过线性变换及偏置操作，将输入映射至 512 维的特征空间，旨在提取更抽象的图像表征。紧随其后，采用 ReLU 激活函数引入非线性，增强模型对复杂数据模式的建模能力，并通过非线性变换进一步提取输入特征的非线性特征，以应对真实世界中复杂的数据分布。接下来，该特征向量被传递至第二个全连接层（fully connected layer2），此层将 512 维特征向量线性映射至 10 维类别空间，为后续的分类决策提供基础。最终，网络通过 Softmax 函数将 10 维的输出向量转化为概率分布，其中每个元素代表图像属于对应类别的概率，从而完成整个分类过程。该模型结构旨在将原始图像像素信息逐步转化为高层语义特征，并最终实现对图像的准确分类。

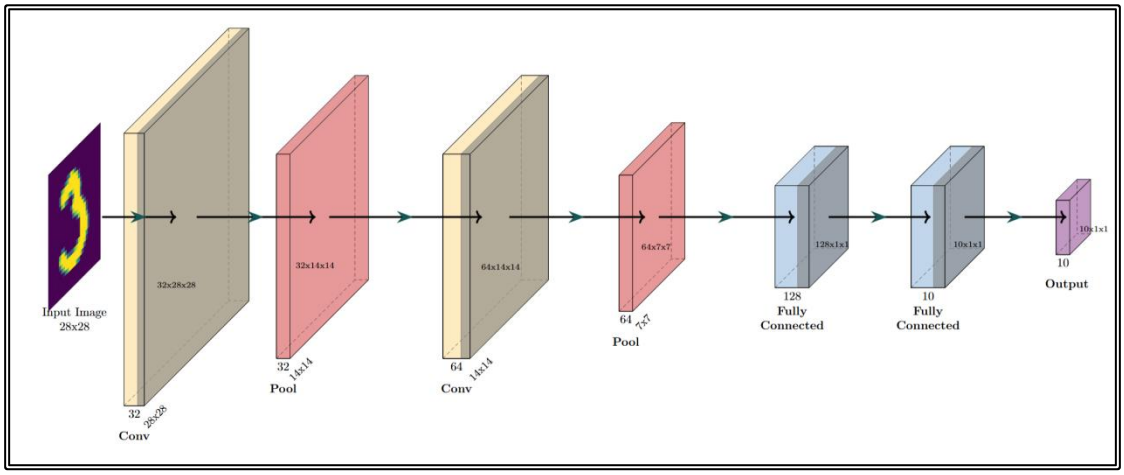


图四 FCN 全连接神经网络之模型架构图

2）CNN 卷积神经网络之模型架构图及相应阐释

该卷积神经网络模型，旨在实现对 MNIST 数据集的图像分类任务。模型首先接收 28x28 像素的单通道图像作为输入，随后，数据被送入第一卷积层（convolutional layer1），该层利用 32 个不同的卷积核对输入图像进行卷积操作，旨在提取图像的局部特征，并将特征图的数量扩展至 32 个通道。此后，采用 ReLU 激活函数引入非线性变换，增强模型的非线性表达能力，并保留卷积操作得到的有效信息。接下来，第一最大池化层（max pooling1）对特征图进行下采样，减小空间尺寸，并保留重要的局部特征信息，以此降低计算复杂度。随后的第二卷积层（convolutional layer2）利用 64 个卷积核对特征图进一步卷

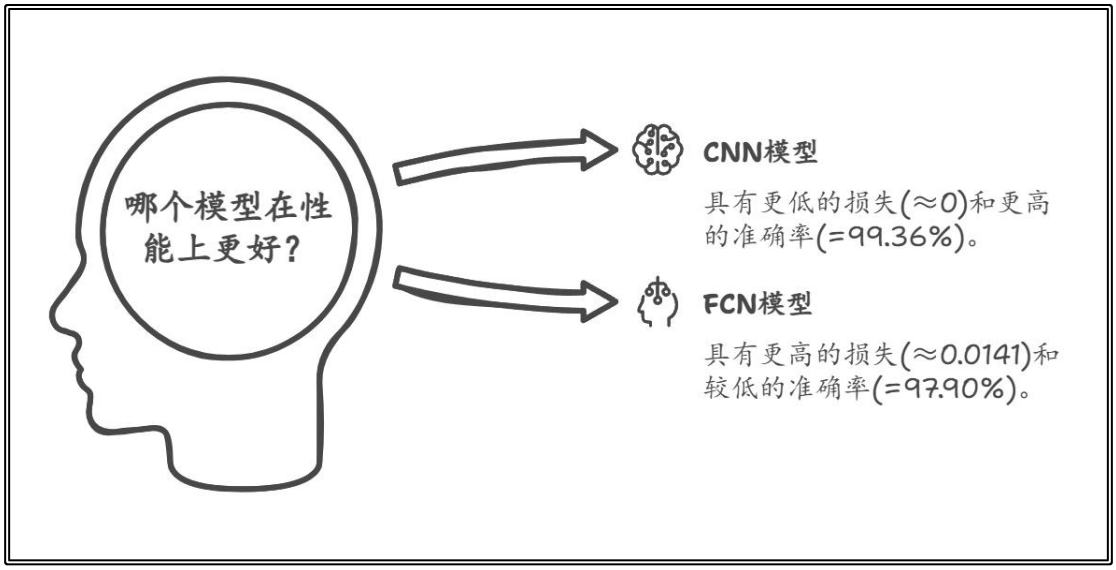
积，旨在提取更深层次的抽象特征，并将特征图的数量扩展至 64 个通道。此后，再次使用 ReLU 激活函数，并进行第二最大池化层（max pooling2）的下采样操作。此时，提取到的特征图被展平为一维向量，以便后续全连接层的处理。该向量被送入第一个全连接层（fully connected layer1），该层通过线性变换及偏置操作，将输入映射至 128 维的特征空间。随后，再次采用 ReLU 激活函数引入非线性。之后，将输出传递至第二个全连接层（fully connected layer2），该层将 128 维的特征向量线性映射至 10 维的类别空间，为最终分类提供决策依据。最后，网络通过 Softmax 函数将 10 维的输出向量转化为概率分布，其中每个元素代表图像属于对应类别的概率，从而完成整个分类过程。该模型结构旨在逐步提取图像的局部和全局特征，并将低级像素信息转化为高层语义特征，最终实现对图像的准确分类。



图五 CNN 卷积神经网络之模型架构图

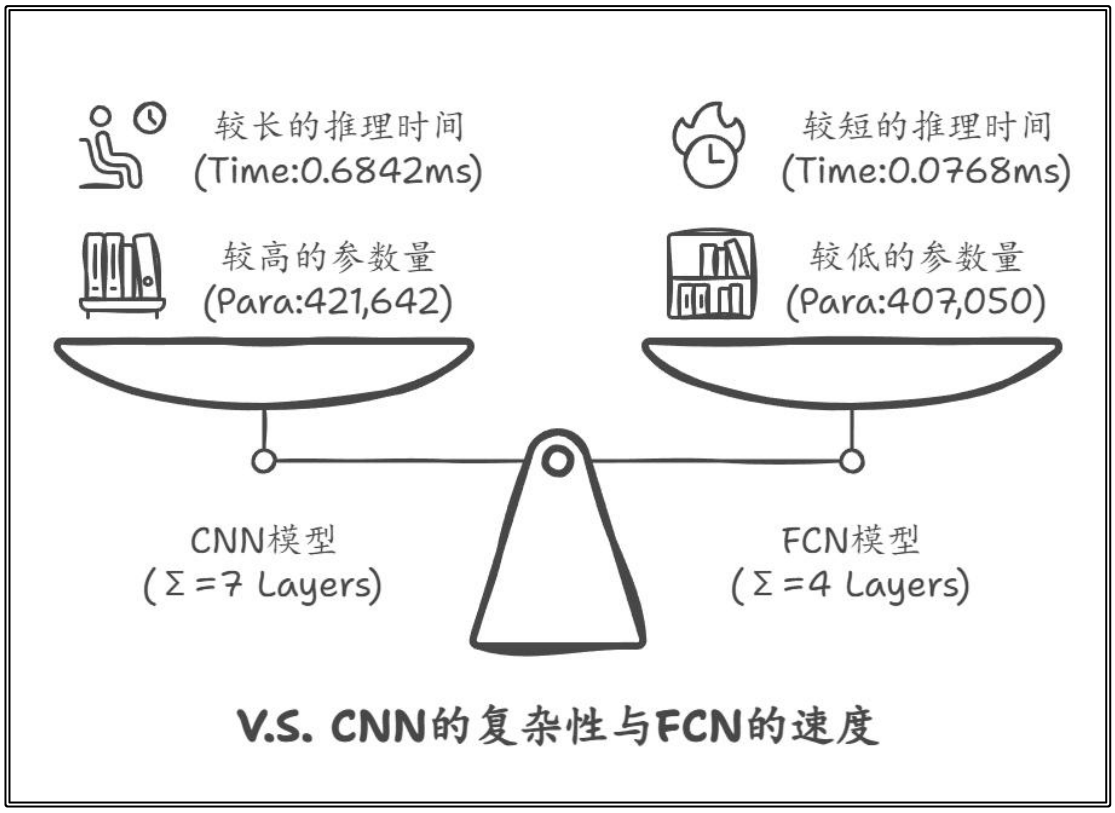
Part3：全连接神经网络 FCN 与卷积神经网络 CNN 训练结果展示

1）FCN 与 CNN 神经网络的训练结果对比展示



图六 FCN V.S. CNN（Facet:平均损失与准确率）

2）FCN 与 CNN 神经网络的模型参数量、推理时间及层数对比展示



图七 FCN V.S. CNN（Facet:平均损失与准确率）

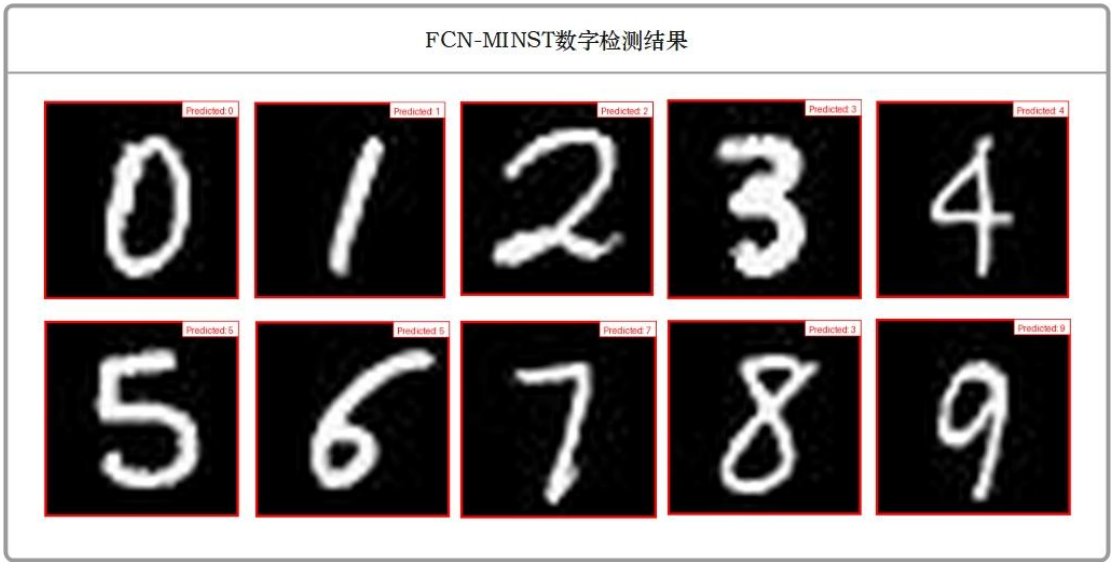
五、项目实现过程二·Flask 调用网络权重文件.pth 的测试页面

Part1:FCN 与 CNN 的数字检测效果

通过对比分析图八(a)和(b)的数据，本文揭示了全连接网络（FCN）与卷积神经网络（CNN）在执行 0-9 数字识别任务时的性能差异。根据验证集上的实验结果，两种模型展现出了显著不同的分类效果。具体而言，FCN 在处理特定数字的分类时暴露出一定的局限性。例如，FCN 倾向于将数字 6 错误地识别为 5，并且经常把 8 误认为 3。此外，该模型对于数字 9 的分类准确性也较为欠缺，暗示其在未来应用中存在潜在的误分类风险。这些发现提示我们，在使用 FCN 进行图像分类任务时，尤其是针对手写体或复杂背景下的数字识别，可能需要额外的优化或调整以提高分类精度。

相比之下，CNN 在同一验证集上表现出色，达到了更高的分类精度。实验数据表明，CNN 能够精确无误地识别从 0 到 9 的所有数字，没有出现任何混淆或误分类的现象。这一优势不仅彰显了 CNN 在模式识别任务中的卓越能力，同时也证明了其独特结构特性——局部感受野、权值共享以及池化层——在图像数据分类任务中的关键作用。

CNN 之所以能够在数字识别方面超越 FCN，部分原因在于它的设计更符合图像数据的空间结构。局部感受野允许每个神经元仅对输入图像的一个小区域作出响应，这有助于捕捉图像中的局部特征；权值共享机制减少了模型参数的数量，同时增强了对不同位置相同特征的检测能力；而池化操作则通过降低特征维度，提高了模型的平移不变性和计算效率。因此，当涉及到图像数据分类等任务时，CNN 相比 FCN 无疑是更为合适的选择。



（a）FCN-MNIST 数字检测结果

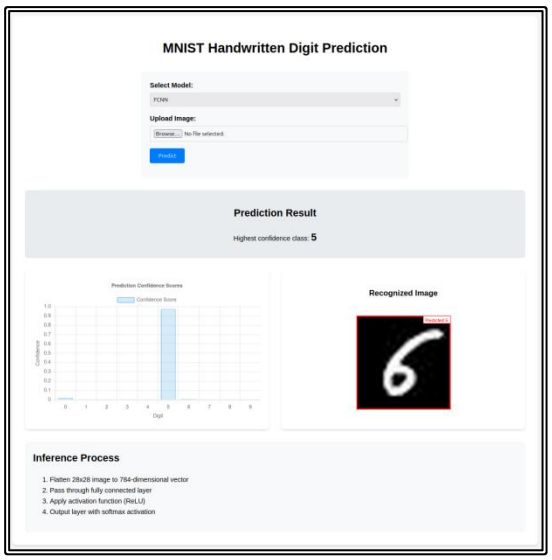
（b）CNN-MNIST 数字检测结果

图八 FCN 与 CNN 在 MNIST 数据集上的识别效果（效果:CNN>FCN）

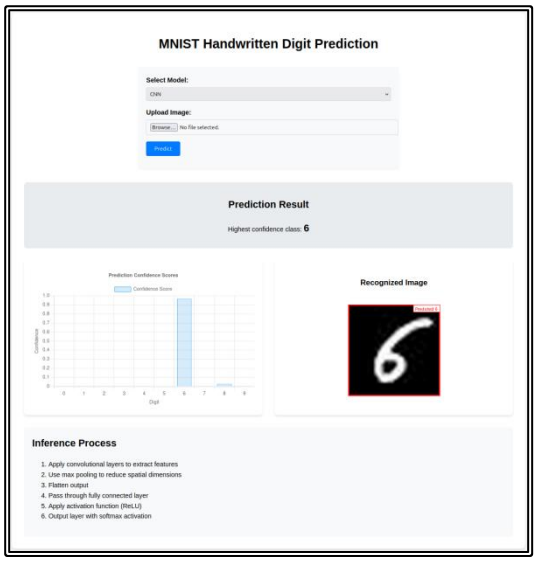
Part2:FCN 与 CNN 的数字检测效果具体对比

在 MNIST 手写数字识别数据集中，卷积神经网络（CNN）的识别效果显著优于全连接神经网络（FCN）。 具体是因为什么呢？实际上是因为两者在架构上存在显著差异，使得 CNN 更能有效地处理图像数据。CNN 的核心优势在于其卷积

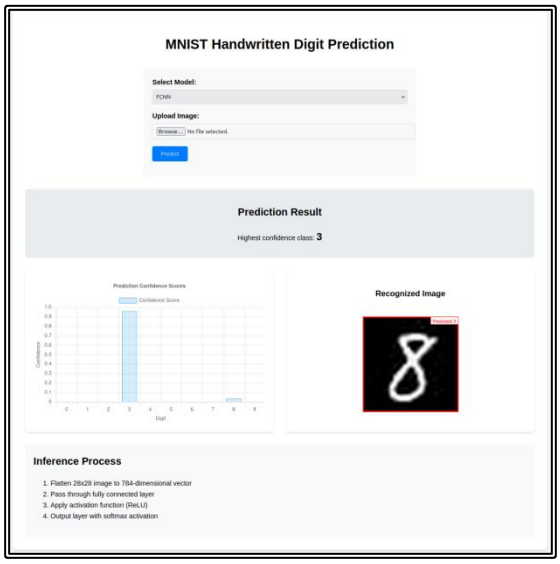
层，它通过局部感受野和权值共享机制，在图像上滑动并提取局部特征，例如边缘、角点等。这些局部特征是识别手写数字的基础，而 CNN 能够有效地捕捉到它们。多层的卷积和池化操作进一步构建了层次化的特征表示，使得网络能够从低级到高级逐步抽象图像信息，逐步降低特征图的空间分辨率，增强模型的平移不变性。相比之下，FCN 直接将 MNIST 图像展平成一维向量，并采用全连接层进行处理，这忽略了图像的二维空间结构，使得模型无法有效地利用像素之间的空间相关性。FCN 的每个连接都有独立的权重，容易过拟合，同时也缺乏局部感知能力，对于位置变化较为敏感。在 MNIST 数据集中，数字的笔画形状和相对位置是重要的识别特征，而 CNN 正好能够有效地利用这些信息。CNN 的局部连接和参数共享策略显著减少了模型参数量，降低了过拟合的风险，同时提高了模型的泛化能力。因此，在 MNIST 手写数字识别任务中，CNN 凭借其更适合图像数据处理的架构，取得了明显优于 FCN 的识别性能，具体对比如下:



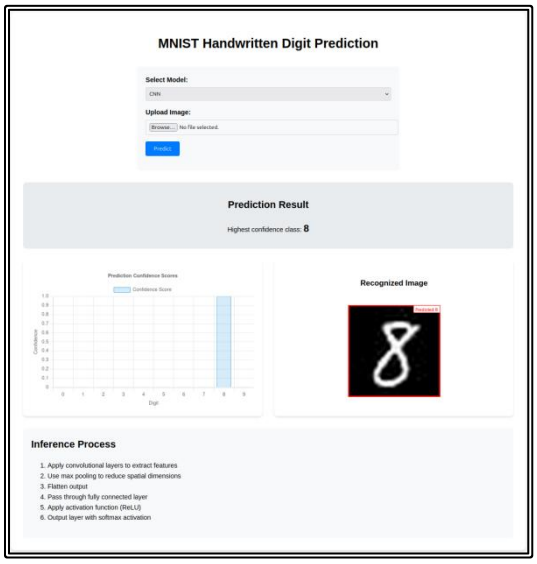
(a) FCN-6



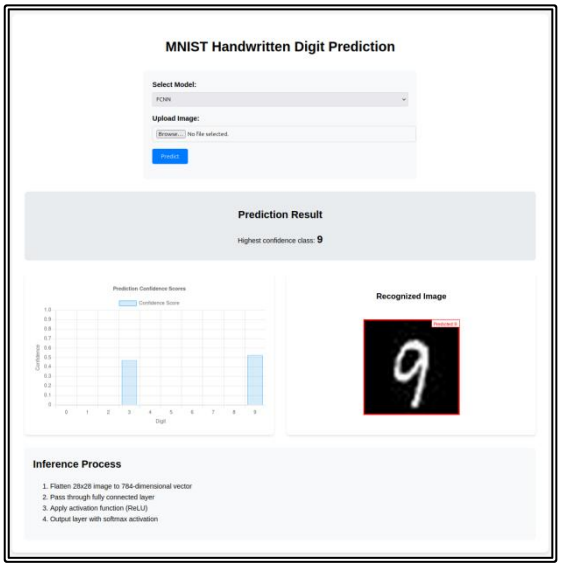
(b) CNN-6



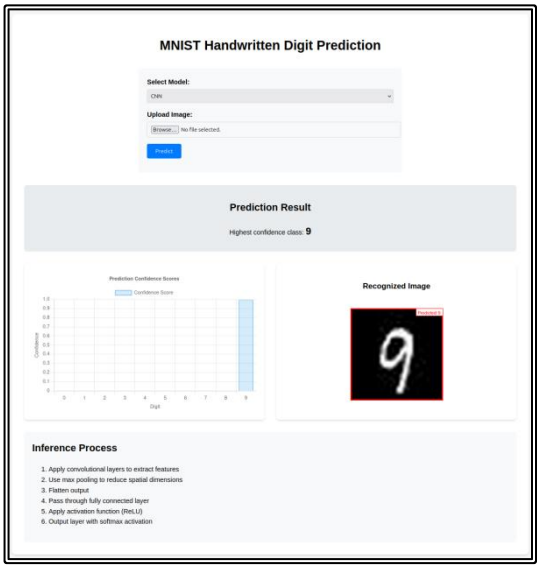
(c) FCN-8



(d) CNN-8



(e) FCN-9



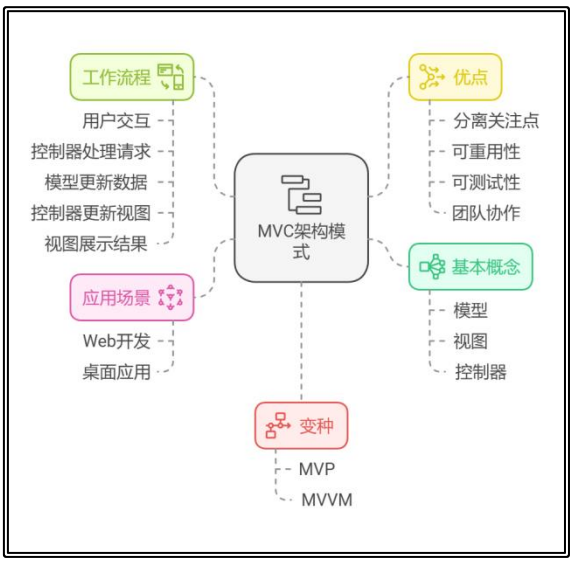
(f) CNN-9

图九 FCN 与 CNN 模型性能测试

六、项目实现过程三·基于 Flask 框架、MVC 架构搭建前后端

Part1:Flask 框架、MVC 架构介绍（Python 环境）

(a) Flask 框架介绍



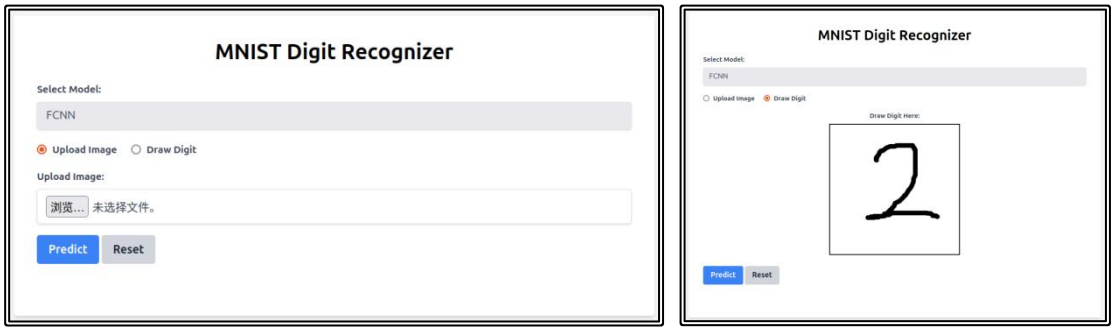
(b) MVC 架构介绍

图十 Flask 框架与 MVC 架构介绍

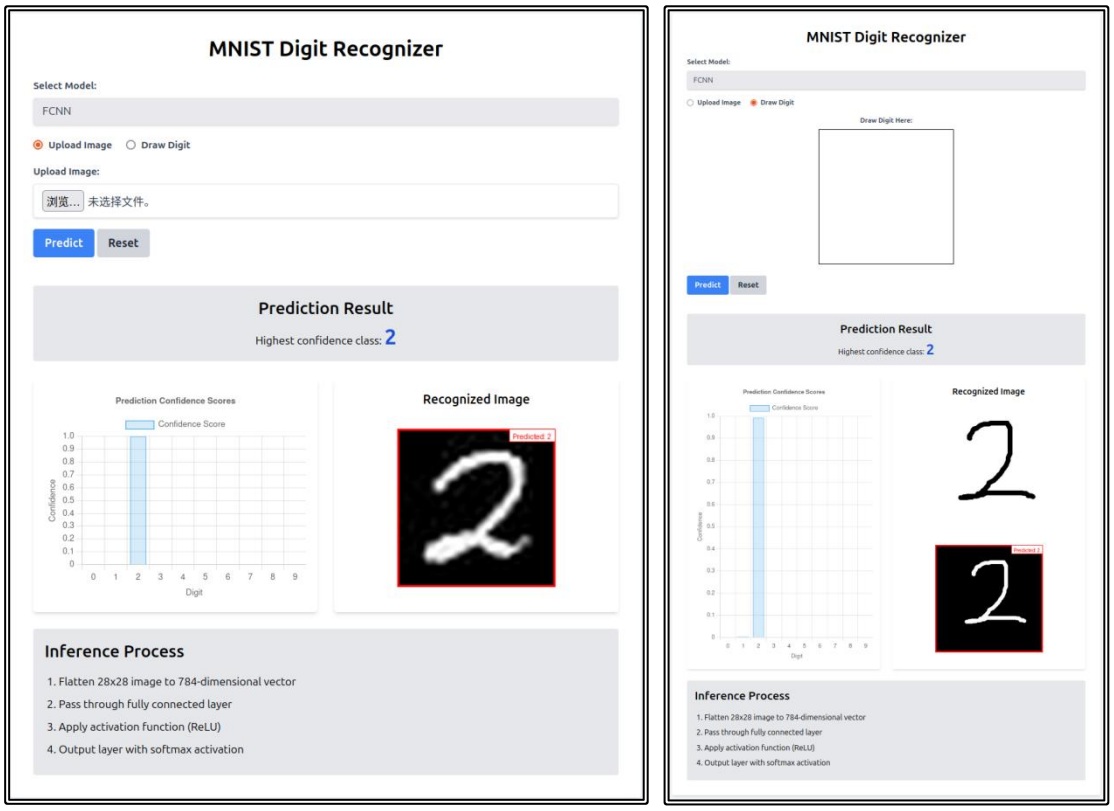
Part2:基于 Flask 框架与 MVC 架构搭建前后端的最终效果

本系统基于 Flask 框架构建了一个前后端分离的手写数字识别应用，采用 MVC 架构模式进行代码组织，其中静态资源存储于 static 文件夹，前端视图由 templates 文件夹下的 HTML 文件定义，后端逻辑处理则集中在 app.py 中。系统实现了用户通过图片上传或手写板绘制两种方式输入手写数字，前端利用

JavaScript 捕获用户输入并编码，通过 HTTP POST 请求与后端 Flask 服务器通信；后端根据用户选择调用预训练的 FCN 或 CNN 模型进行数字识别，并将结果以 JSON 格式返回至前端；前端接收数据后，解析并在用户界面呈现识别结果，包括置信度得分和可视化表示，此系统不仅实现了用户界面的交互功能，也有效解耦了前后端逻辑，具备良好的可维护性和扩展性。



(a) 图片上传与画布上传功能首页



(b) 图片上传与画布上传功能展示页

图十 Flask 框架与 MVC 架构介绍

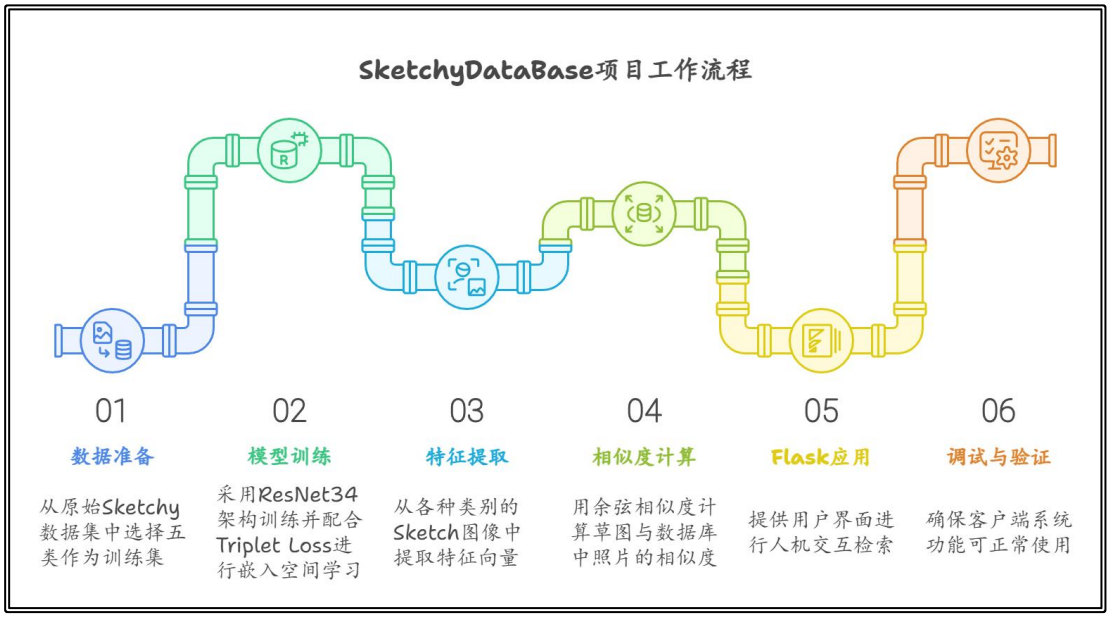
温馨提示：基于 Python-Web 手写数字识别系统的完整开发过程展示已结束！

七、项目实现过程四·基于手绘交互的图像检索功能复现

Part1:复现思路设计展示

本文首先克隆了 SketchyDatabase 项目中的数据集到本地电脑中，并根据 requirements.txt 文件安装了所有必要的库，以确保版本兼容性。接着，本文在数据预处理阶段，检查了图像变换步骤，包括缩放、裁剪和归一化，以确保其正确实现。由于数据量较大，本文参考了老师上课提出的建议——通过筛选对现有数据集进行缩小。因此，本文以类别为标准，从原始数据集中，选择了五类，并对缩小后的数据集进行了训练集和验证集及测试集的分割。

在模型训练阶段，本文参考 SketchyDatabase 项目中所采用的 ResNet 架构，并以 ResNet34 为骨干网络，加载该预训练权重进行模型训练。对于手绘交互部分，我们设计并集成了一个手绘输入界面，使用 matplotlib 库来获取和处理手绘图像的像素数据（与手写数字识别系统一致）。最后，在图像检索阶段，本文从数据集中筛选出与当前手绘图像相同类别的图像，并将其展示给用户。



图十一 SketchDatabase 复现思路设计展示

Part2:SketchDataBase 复现过程展示

本项目旨在利用 Sketchy 数据库进行人机交互任务，重点是匹配来自五个不同类别的草图和照片。dataset_processor.py 脚本在数据预处理和对草图-照片对的对齐中起到了至关重要的作用，为模型训练创建了结构化的数据集。对于实际的模型训练，train_sketchdb_model.py 采用了 ResNet-34 架构，结合 Triplet Loss 损失函数和 SGD 优化器，以捕捉草图和相应照片之间的语义关系。然而，模型的表现未达到预期，可能是由于数据集本身的挑战，或架构在这一特定任务上的适应性问题。在前端，templates/index.html 提供了直观的用户界面，方便用户上传并交互使用草图和照片输入。后端逻辑通过 app.py 进行协调，

连接用户界面与模型的预测过程。该设计不仅旨在完善基于草图的检索系统，也凸显了 SketchyDataBase 在更广泛的人机交互应用中的潜力，特别是在需要从非写实绘画中理解视觉信息的领域。

图十二 SketchDatabase 工作流程

Part3:SketchDataBase 结果说明与展示

本项目展示了基于草图的图像检索系统，通过上传或手绘草图进行图像搜索。系统首先通过提取用户输入的草图特征，并与数据库中的图像特征进行比对(参照 Github 中 SketchyDataBase，本文使用余弦相似度进行度量草图特征与数据库图像特征两者间的相似度—Epochs:100； batch_size=32,lr:0.001；margin=1.0；p=2)，返回最相似前六个的图像。在测试示例中:

在处理上传的飞机草图时，系统成功地返回了与查询图像高度相似的三张飞机图像，也产生了三张误检图像，表明 Resnet-34 对该类别的检测不够敏感。而对于闹钟类别，出现的误检现象更为严重，表明 Resnet-34 仍需进一步改进。

而对于手绘的时钟和大猩猩的草图，Resnet-34 在处理时钟草图时，同上传的飞机草图和闹钟草图类似，均存在误检现象，正确率≤50%。

通过以上实验结果可得：已经过权重训练的 Resnet-34 架构的模型性能仍不足以满足 SketchyDataBase 的建设，但总体上展示了其在草图基础图像检索任务中的可行性，具有进一步提升的潜力。

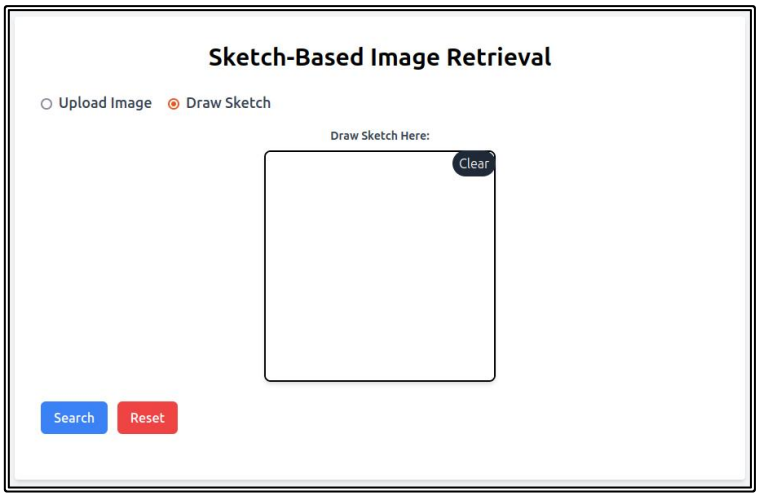
· 展示一——常见 Resnet 网络汇总（From Resnet-18 To Resnet-152）

Layer	Output size	ResNet-18	ResNet-34	ResNet-50	ResNet-101	ResNet-152
Conv1	112×112	7×7, 64, Stride 2	3×3, 64, Stride 2	3×3, 64, Stride 2	3×3, 64, Stride 2	3×3, 64, Stride 2
Conv2_x	56×56	[3×3, 64; 3×3, 64] ×2	[3×3, 64; 3×3, 64] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3
Conv3_x	28×28	[3×3, 128; 3×3, 128] ×2	[3×3, 128; 3×3, 128] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×8
Conv4_x	14×14	[3×3, 256; 3×3, 256] ×2	[3×3, 256; 3×3, 256] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×23	[1×1, 256; 3×3, 256; 1×1, 1024] ×36
Conv5_x	7×7	[3×3, 512; 3×3, 512] ×2	[3×3, 512; 3×3, 512] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3
Output	1×1	Average pool, 1000-d fc, Softmax	Average pool, 1000-d fc, Softmax	Average pool, 1000-d fc, Softmax	Average pool, 1000-d fc, Softmax	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

图十三　常见 Resnet 网络汇总（MadeBy：Latex-Overleaf）

· 展示二——用户端使用首页

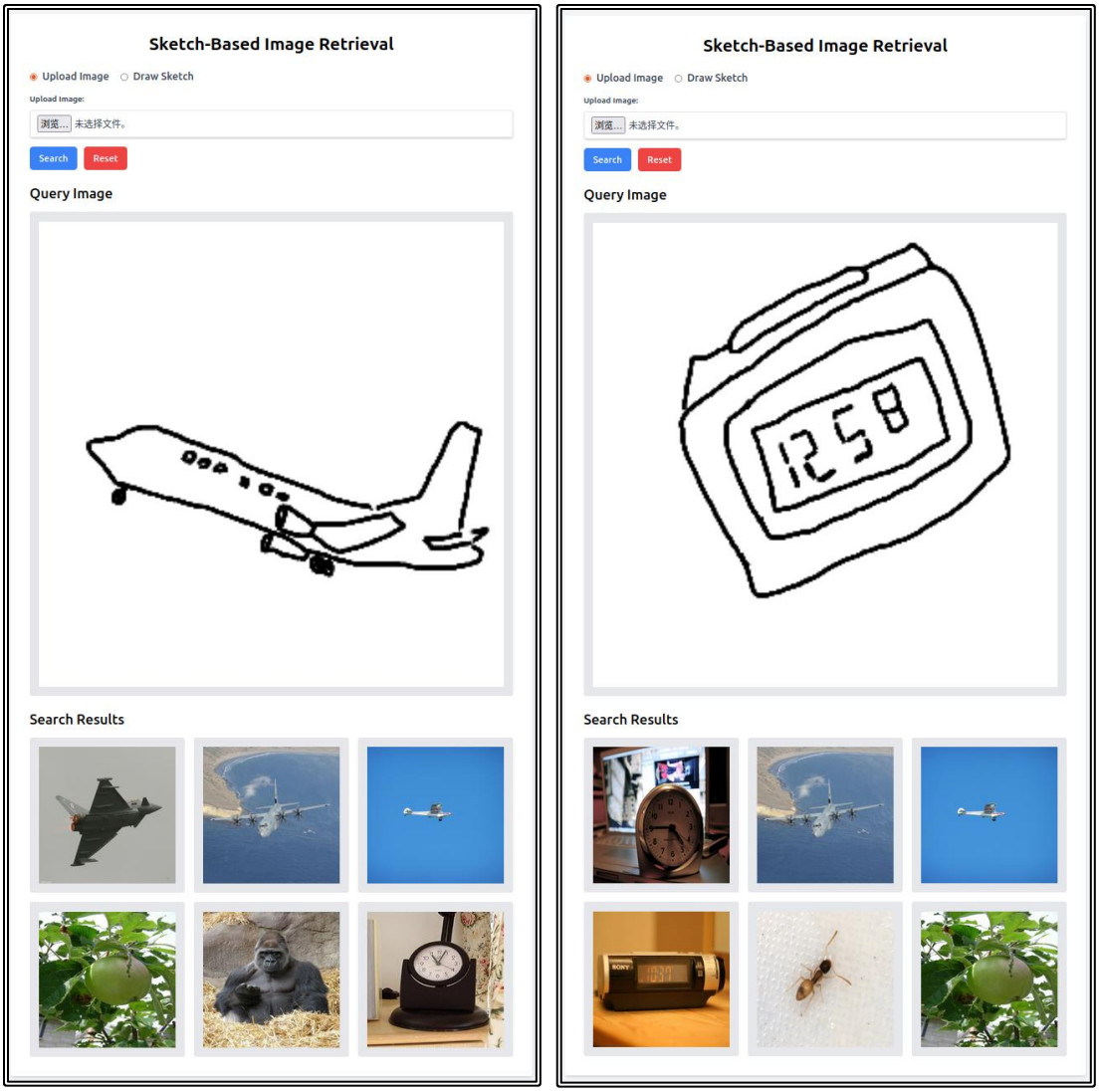
（a）图片上传功能首页



（b）手绘画板功能首页

图十四 图片上传与手绘画板功能首页

·展示三——用户端上传功能测试（以飞机为例）



（a）上传飞机测试图片一　　（b）上传飞机测试图片二

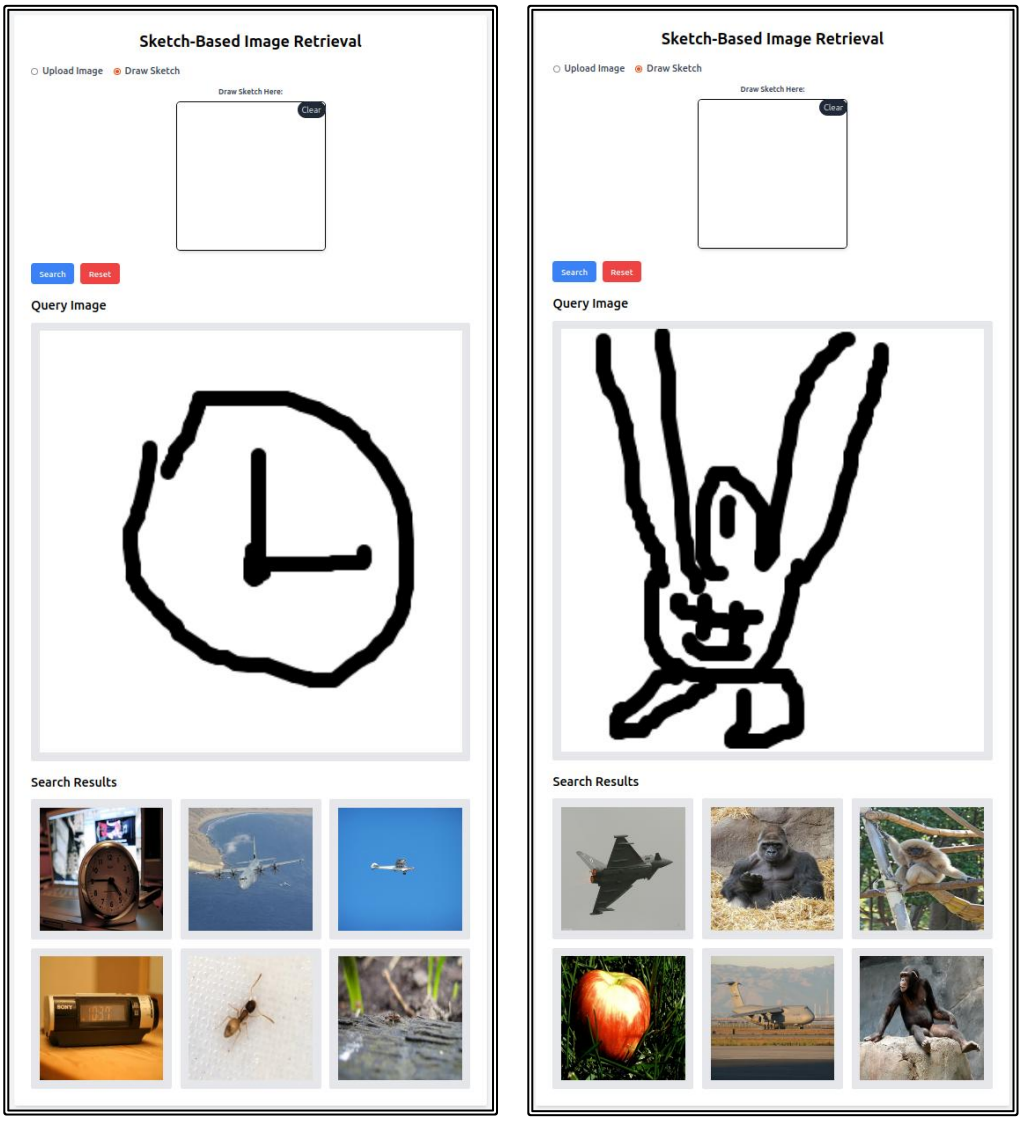
图十五 Sketchy DataBase 效果测试—上传手绘图像（Search Results=6）

注：

1）数据集包含五个类别：飞机、警钟、苹果、猿猴、蚂蚁，共计 4011 张图像。

2)本文采用Resnet-34 网络的原因是:RTX3060 显卡算力无法支持Resnet-50 网络，故本文无法选用更高精度的模型。（据 Github 原作者结论，两者效果相差甚远，原作者也无法解释！这也就是神经网络难以解释而又魔幻的魅力吧！）

·展示四——用户端上传功能测试（以时钟和大猩猩为例为例）



图十六 Sketchy DataBase 效果测试—画板手绘（Search Results=6）

八、项目完成情况

1、利用脚本语言按照界面设计的基本原则设计画板页面。（20 分）✔

2、基于 Flask 项并按照 MVC 原则开发手写数字识别项目，分析采用卷积和全连接方法对结果的影响（运行效率、参数量及层数方面分析）。（30 分）✔

3、基于手绘交互的图像检索代码复现（20 分）✔

4、项目实验结果可视化展示（20 分）✔

注：1、2、3、4 均已详细且充分地实现！感谢汪飞老师的评阅！祝好！