

成绩：

**方向课综合训练报告**

题　　目：基于ViT网络的手写数字字符图像的检测与识别

班　　级： 计算机22-4班

学　　号： 2204010429

姓　　名： 郑朝旭

指导教师：　　　　　 于林森（14组）

系主任： 李成严

计算机科学与技术学院

2025年6月21日

**基于VIT网络的手写数字字符图像的检测与识别**

1. **题目分析**

在传统的手写字符识别任务中，常使用卷积神经网络（CNN）对标准化图像（如 MNIST）进行训练与分类。但在实际应用中，原始图像往往存在字符粘连、倾斜、光照不均、背景干扰等问题，且字符排布格式多样化，严重影响识别精度。为此，本系统在输入图像质量无法保证的前提下，从图像处理角度进行预处理优化，再结合 Vision Transformer（ViT）模型的全局建模能力，实现对手写字符的准确识别。

为实现较为完备的识别系统，我将题目分解为以下几个核心子任务进行实现：

**1.1字符区域定位**

首先对输入图像进行结构化分析，定位包含手写数字的区域，通常是一整行连续数字。该步骤结合边缘检测、投影分析或连通域分析等方法，初步框定感兴趣区域（ROI），排除图像中无关区域与干扰。

**1.2字符与背景分离**

在获得ROI区域后，需对图像进行去噪与自适应二值化处理，将数字前景与背景有效分离，生成高对比度的 mask 图像。本项目使用 Sauvola 局部阈值方法以适应光照变化，同时控制形态学参数，保证数字主体完整保留。

**1.3图像几何矫正**

为消除字符行的倾斜与变形问题，提高分类器预测的准确率，需要对 mask 图像进行几何归一化处理，包括角度旋转与比例校正，使其接近水平对齐，便于后续字符切分。

**1.4单字符提取与规范化裁剪**

对校正后的图像进行字符级分割，通过连通域分析与形态筛选提取出每个独立字符，随后统一裁剪为 28×28 像素灰度图，与标准 MNIST 图像格式一致，作为 ViT 分类器的输入。

**1.5基于ViT的字符识别**

将上述预处理后得到的标准化图像输入 Vision Transformer 网络进行分类预测。该模型实现参考了论文源代码，后经过自主修改适配并训练，具备较强的空间建模与全局语义理解能力，适用于字符形态多样的识别场景。最终，系统输出每个字符的识别标签，并在 mask 图像中对其位置与预测结果进行可视化标注。

综上所述，本手写数字图像检测与识别系统的总体框架图如**图1**所示。

图示

AI 生成的内容可能不正确。

图1基于ViT的手写数字图像检测与识别系统整体框架图

**2.整体设计**

**2.1文本行检测模块**

首先，系统需要定位输入图像中包含手写数字的感兴趣区域（ROI）。通常这些手写数字沿水平方向成行排列，因此本设计引入文本行检测模块来自动找到每一行手写数字所在的位置。通过该模块，可以将原始图像中可能的无关背景区域排除，只保留含有数字的行区域，降低后续处理的干扰和计算量。本系统采用了深度学习OCR检测算法（如 PaddleOCR 提供的文本检测模型）来完成此任务。该检测器能够从图像中检测出每行文字的轮廓位置，并输出每个文本行的外接边界框。经过这一阶段处理，系统获得了若干经精确定位和裁剪的数字行图像，为后续字符级处理提供了可靠输入。

**2.2字符与背景分离模块**

获得行级ROI图像后，下一步是将字符前景从背景中分离出来，即前景提取。由于手写图像往往存在噪声、不均匀光照等问题，需要进行必要的预处理以提高清晰度。首先对行图像进行去噪处理（如平滑滤波）和灰度化，减轻背景杂讯的影响。随后，采用自适应阈值法对图像进行二值化，将手写数字前景与背景有效分离，生成高对比度的二值掩膜(mask)图像。相比固定阈值，自适应局部阈值（例如 Gaussian 算法）能够根据局部光照情况动态确定阈值，保证在光照不均的情况下仍能正确区分前景和背景。在获得初步的二值图后，将进一步应用一系列形态学操作来优化前景：适度“填充”断裂的笔画、连接因噪声导致的细小间隙，并去除孤立的噪点。最后还对字符轮廓做轻微的膨胀处理，以确保数字笔画的完整性。经过此模块处理，输出的是一个干净的二值化字符图像，其中前景为清晰的手写字符、背景为纯黑色，为后续精确的字符切分提供了保障。

**2.3图像几何校正模块**

在二值化后的掩膜图像中，手写字符行可能会存在一定的倾斜或弯曲。这种几何畸变不利于后续字符的切分和识别。为提高识别精度，本系统设计了图像几何矫正模块，对整行字符进行倾斜校正和形变调整。具体而言，系统会分析二值图中字符前景像素的分布，计算出整行文字相对于水平线的倾斜角度，并据此对图像进行旋转校正。此过程可采用基于主成分分析（PCA）的算法来完。PCA能够找出前景像素分布的主方向，该方向即对应文字行的倾斜方向。将此主方向旋转对齐到水平轴，即可实现对字符行的去倾斜校正。经过几何矫正，字符行被规范化为接近水平，字符间距和比例也得到适当调整，便于后续的逐字符切分处理。

**2.4字符检测分割模块**

完成行级的定位、清理和校正后，系统进入字符切分提取模块，将每一行中的各个手写数字单独分割出来。首先，基于前一步得到的规整二值图，系统通过连通域分析来检测图像中的连续前景区域，每个连通区域对应一个字符候选。由于某些手写数字可能彼此相连或笔画粘连，本模块还结合投影分析和聚类等策略对可疑区域进行进一步切分：例如，当检测到某个连通域的宽度明显大于高度（预示可能包含两个紧挨的数字）时，可通过垂直方向的投影直方图寻找间隙谷值来尝试将其一分为二；若投影法未能有效分离，则可改用对该区域前景像素的坐标进行K-means聚类，将像素点自动分成两簇，从而确定可能的切割点。经过多策略处理后，模块会得到一系列分割后的候选字符区域。

在获得初步的字符候选区域后，系统对这些区域的属性进行分析筛选。根据连通域的面积、像素密度、长宽比等统计特征，过滤掉过小的噪声块或明显不属于字符的区域。同时，如果某个小区域完全嵌套在另一个大区域内部，则判定其为冗余的重复检测，予以去除。对于仍然过宽的候选框（可能仍包含两个字符的情况），则再次采用精细的投影谷值定位和连通性检查进行二次切分，确保将相邻字符彻底分离。经过上述处理后，剩余的有效字符边界框按照横坐标位置从左到右排序，对应于原图中字符的阅读顺序。

最后，针对每一个确定的字符区域，系统提取其中的前景像素并进行进一步的规范化处理。为了提高分类器识别的准确率，需要尽可能提供姿态端正且无多余噪声的字符图像。为此，首先对每个字符连通区域仅保留最大的连通成分（去除内部碎片噪点），并计算该区域的二阶矩以获取字符主轴方向，对字符图进行微调旋转使其纵向对齐；然后应用形态学闭运算填充字符内部可能存在的小孔洞，并再做一次轻微膨胀使笔画达到适当的粗细；接着对字符图像执行高斯平滑去除尖锐噪声，随后重新二值化以得到清晰边缘。最后将得到的字符图裁剪并缩放规范为28×28像素的灰度图像，统一尺寸和格式（与MNIST等标准数据集一致），作为后续ViT分类器的输入。至此，系统完成了从整行图像中检测并提取单字符图像的流程。

**2.5字符识别模块**

在完成字符分割与预处理后，系统进入**字符识别模块**。该模块采用基于Vision Transformer (ViT)的深度学习分类器对每个单独的字符图像进行识别分类。首先，将规范化后的字符小图输入ViT模型，由模型提取图像的深层次特征并进行类别判别。ViT模型具有强大的全局建模能力，能够有效捕获手写数字的整体笔画结构和局部细节，对各种书写风格的字符都具有鲁棒的识别性能。本系统所用的ViT分类器架构参考了相关论文的开源实现，并根据手写数字识别任务进行了适配和训练。最终，分类器输出每个字符所属的数字类别标签。为了直观展示识别结果，系统还会将识别出的数字标签与原始图像对应位置关联，形成带有识别标注的结果输出。例如，对于输入图像中的每一行数字，系统给出按顺序识别的数字串，同时在图像上标记出各数字的位置和识别值。经过以上各模块的协同工作，整套手写数字识别系统即可对复杂场景下的手写数字进行准确识别。

**3.具体实现**

**3.1实现字符所在行定位**

本系统使用了PaddleOCR模型实现字符所在行的定位。PaddleOCR是百度飞桨团队开发的一款开源的OCR工具库，具备高性能、轻量化和易于部署的特点。它提供了一套完整的文本处理流程，包括文本检测、识别和版面分析，已被广泛应用于自然场景、票据扫描、文档识别等实际任务中。PaddleOCR支持DBNet、EAST、SAST、PSE等多种主流的文本检测算法，并配备丰富的预训练模型，极大地降低了使用门槛。

DBNet（Differentiable Binarization Network）是PaddleOCR提供的一种高效的文本检测算法，以良好的性能和实时检测能力而闻名。DBNet的主要创新在于使用了一个可微的二值化模块（Differentiable Binarization Module），让网络自动学习出最佳的文本区域划分阈值，从而实现准确的文本检测。DBNet由四个核心部分构成：

**3.1.1骨干网络(Backbone)**

通常使用轻量级的网络结构如MobileNetV3或ResNet50，其作用是从输入图像中提取出丰富的、多尺度的特征。这些特征能够很好地适应各种复杂场景下的文本检测需求

**3.1.2特征融合模块(FPN)**

将骨干网络提取的多尺度特征进行有效融合，生成统一且更加全面的特征图。该模块可以有效提升模型对于不同尺寸和不同方向文本的检测能力。

**3.1.3头部网络(Head)**

主要由一系列卷积层与转置卷积层构成，用于生成文本区域的概率图和阈值图。概率图描述每个像素属于文本区域的可能性，而阈值图则用于确定每个像素自适应的二值化阈值。

**3.1.4可微二值化模块(DB Module)**

这是DBNet算法最核心的创新部分。通过结合概率图与阈值图，DB模块能自动学习并确定最佳的二值化阈值，将概率图转化成清晰准确的二值图，精确地标定出文本区域的边界。

DBNet网络框架图如**图2**所示。

图示

AI 生成的内容可能不正确。

图2 DBNet网络框架图

基于DBNet的文本检测流程包括图像预处理、特征提取与融合、文本区域检测和后处理四个主要步骤。输入图像首先进行缩放、归一化和填充，以符合网络输入要求；随后骨干网络和特征融合模块提取并融合图像特征；接着头部网络和可微二值化模块生成文本区域概率图；最后通过连通区域分析确定最终的文本边界框。DBNet算法在ICDAR2015数据集中表现突出，精度达到92.3%，召回率达到85.6%，并且具备极佳的实时性（在NVIDIA V100 GPU上推理速度达到12FPS），部署便捷，广泛适用于各种复杂场景的文本检测任务，具有较强的实际应用价值。

在代码实现时，只要调用PaddleOCR即可使用。

**3.2实现字符与背景分离**

**3.2.1基于高斯加权法的自适应阈值分割**

由于图像中存在明显的光照梯度、阴影或局部亮度不均，全局阈值（Otsu）往往在某些区域过饱和而在另一些区域过分空白。为了解决这一问题，可以采用自适应阈值法。其为每个像素动态计算局部阈值。

本系统中使用了高斯加权法实现自适应阈值分割。用与邻域中心距离相关的高斯权重计算加权均值：

**:**表示当前像素在图像中的行列坐标

以为中心的局部邻域大小为的方形窗口。在该邻域内，对所有 像素进行加权平均。  
 **:**高斯权重函数，表示以为相对偏移量时，对应邻域中心的权重值:

在邻域中，位于位置的像素灰度值

由于引入了高斯函数，所以这种方法更平滑、抗噪声。利用高斯加权法在光照不均的图像上阈值分割1结果如**图3**所示。

文本

AI 生成的内容可能不正确。

图3高斯加权法阈值分割结果

但我们发现，只利用自适应阈值分割往往是不够的。虽然提取出了前景，但是仔细看还是存在了许多问题。首先是**笔画断裂**，数字3、7的弧线被黑缝所切断，极细笔画处甚至只有一个像素宽，这会导致连通域把同一个数字分为两块，后续投影、K-means误切。其次是**横向裂缝**，斜写导致横画更靠边，局部光照又弱，出现窄而长的水平黑带。这会导致数字横画被分割成两段，分割时可能把同一字符拆分成上下两块。然后是**孤立白点**，这是阈值化后纸屑、灰尘留下来的零散白点。这会导致连通域额外产生几个候选框，后面要手动过滤，失去了自动检测的意义。最后，数字的笔画普遍偏细，这在连通域检测或者几何矫正时因为插值出现锯齿。为此，我们需要引入形态学操作来修补这些小问题。

在图像处理领域，形态学是一种基于形状的图像分析技术，用于提取和处理图像的形态特征。这包括**膨胀**、**腐蚀**、**开运算**、**闭运算**等操作，广泛应用于边缘检测、图像分割和噪声去除等方面。形态学操作的核心概念是结构元（Structuring Element），这是一个定义形态学操作如何应用于图像的形状或模板。结构元可以是任意形状的，如方形、圆形、十字形等，应用于图像时，结构元素在图像上滑动，决定了哪些像素会被膨胀、腐蚀或其他形态学操作影响。通过组合这些基本操作，可以实现复杂的图像处理任务，如物体边界提取、骨架化、形状分析等。形态学在许多图像处理应用中具有重要作用，尤其是在需要基于形状特征进行分析的场景中。

**3.2.1腐蚀操作**

腐蚀是最基本的形态学操作之一，它能够将图像的边界点消除，使图像沿着边界向内收缩，也可以将小于指定结构体元素的部分去除。腐蚀用来“收缩”或者“细化”二值图像中的前景，借此实现去除噪声、元素分割等功能。腐蚀操作前后对比图如**图4**所示，左图是原始图像，右图是对其进行腐蚀操作的处理结果。

图片包含 QR 代码

AI 生成的内容可能不正确。

图4腐蚀操作前后对比

在腐蚀过程中，通常使用一个结构元(Kernel)来逐个像素地扫描要被腐蚀的图像，并根据结构元和被腐蚀图像的关系来确定腐蚀结果。结构元可以自主设计，其与原图像的计算方式与卷积类似，都是对应位置相乘，然后用累加值覆盖最中间的值，也就是每次判定的点都是结构元中心所对应的点。腐蚀结构元遍历过程如**图5**所示。

电子计算机

AI 生成的内容可能不正确。

图5腐蚀结构元遍历过程

也就是说，结构元覆盖的区域只有是全白(1)，其结构元中心才是白，否则会被腐蚀成黑(0)。

**3.2.3膨胀操作**

膨胀操作是形态学中另外一种基本的操作。膨胀操作和腐蚀操作的作用是相反的，膨胀操作能对图像的边界进行扩张。膨胀操作将与当前对象（前景）接触到的背景点合并到当前对象内，从而实现将图像的边界点向外扩张。如果图像内两个对象的距离较近，那么在膨胀的过程中，两个对象可能会连通在一起。膨胀操作对填补图像分割后图像内所存在的空白相当有帮助。膨胀操作前后对比图如**图6**所示，左边是原始图像，右边是对其进行膨胀操作的处理结果。

QR 代码

AI 生成的内容可能不正确。

图6膨胀操作前后对比

同腐蚀过程一样，在膨胀过程中，也是使用一个结构元来逐个像素地扫描要被膨胀的图像，并根据结构元和待膨胀图像的关系来确定膨胀结果。膨胀结构元遍历过程如**图7**所示。

日历

AI 生成的内容可能不正确。

图7膨胀结构元遍历过程

可以看出，只要结构元覆盖的区域有白色(1)，那么这一块结构元中心就会变成白色,否则仍然是黑(0)。

**3.2.4开运算**

开运算进行的操作是先将图像腐蚀，再对腐蚀的结果进行膨胀。开运算可以用于去噪、计数等。开运算前后对比图如**图8**所示，左边是原始图像，右边是对其进行开运算的处理结果。在本示例中，其结果与腐蚀结果相同

QR 代码

AI 生成的内容可能不正确。

图8开运算前后对比

**3.2.5闭运算**

闭运算是先膨胀、后腐蚀的运算，它有助于关闭前景物体内部的小孔，或去除物体上的小黑点，还可以将不同的前景图像进行连接。开运算前后对比图如**图9**所示，左边是原始图像，右边是对其进行开运算的处理结果。

QR 代码

AI 生成的内容可能不正确。

图9闭运算前后对比

在示例中，二值图里噪声是白点（前景），背景是黑底。我们相当于对整个白色区域做了闭运算，就把这些孤立的白点都膨胀并连在一起，看起来就像噪声被提取了。

**3.2.6形态学操作修补**

有了上述形态学操作知识，我们可以综合运用形态学操作来解决问题。对于笔画断裂问题，我们可以采用核为的闭运算把小于等于2px的缝隙糊平，笔画重新连成一体，运用闭运算结果如**图10**所示。

文本

AI 生成的内容可能不正确。

图10运用闭运算修复笔画断裂

接着对于横向裂缝问题，运用核为的水平闭运算，保证只在水平方向上膨胀，再腐蚀，专门封住黑缝，运用闭运算结果如**图11**所示。虽然肉眼看不出来，但是为了像素级别的稳妥，还是要做。

文本

AI 生成的内容可能不正确。

图11运用闭运算修复横向断裂

然后为了消除孤立的白色噪声点，采用核为的开运算把面积小于4px的噪声先腐蚀掉，再填回主体笔画。运用开运算结果如**图12**所示

文本

AI 生成的内容可能不正确。

图12运用开运算消孤立白点

最后为了加强笔画，可以使用核为的轻度膨胀把所有白色笔画加粗1px。运用腐蚀结果如**图13**所示

文本

AI 生成的内容可能不正确。

图13运用腐蚀增强笔画

**3.3实现图像几何矫正**

得到二值mask之后，我发现它是倾斜的，这不利于后续的字符提取以及分类器的检测，因此我们需要尽可能地进行几何矫正，使得每个字符尽量水平。我在几何校正模块采用了一种基于主成分分析（PCA）的倾斜校正方法。能够使用主成分分析是因为我发现所有手写数字都在同一行上，这些数字的像素分布在空间会呈现一个近似共线、共向的主方向。

首先从二值文字掩码中提取所有前景（字符）像素的坐标分布，然后通过主成分分析找出这些像素分布的主方向，将此方向与水平轴对齐即可完成去斜处理。接下来描述 PCA 提取主方向的流程。(先验知识，字符在一行上)

首先对掩码执行像素阈值提取，得到所有非零像素的行列坐标，并将其组合为的坐标矩阵；接着计算坐标矩阵的均值和特征向量，其中第一个特征向量对应最大方差方向，也就是文字行的主方向；然后利用：

得到该方向相对于水平轴的倾斜角度，再将其转换为度数；最后以图像中心为旋转中心，构造旋转矩阵并对掩码做逆向旋转，插值方式选用最近邻并用零值填充空白，便能将文字精准校正到水平状态。

**3.4实现字符检测定位**

几何矫正过后，我通过连通域检测实现字符检测定位。在二值图像中，连通域指的是所有互相连在一起的前景像素的集合。我们假设二值图像。其中表示前景，代表我们感兴趣的对象(也就是我们要检测的手写数字)；代表背景。若两个像素之间存在一个由4-邻域(上下左右)或者8-邻域(4邻域基础上加上4个对角方向)相邻像素构成的路径，就称它们在同一个连通域内。

连通域检测通常采用两遍扫描算法配合并查集完成连通域标记。对于第一遍扫描，按行主序遍历每个像素，若为前景，则考察其上、左（及对角）已经打标记的邻居，若邻居标签集为空，新建标签；若不为空，取最小标签并登记所有邻居标签的等价关系。之后通过并查集在线合并等价标签，保证后续查询能找到同一连通域的代表。对于第二遍扫描，再次遍历，将每个前景像素的标签替换为其并查集根代表，实现标签压缩和去重。

当我们直接使用连通域检测，其检测结果如**图14**所示。

图示

AI 生成的内容可能不正确。

图14直接使用连通域检测结果

可以发现直接使用连通域不仅检测到了所有的小噪声，而且还两个相近的字符当成一个整体进行检测了，例如35和23，因此光进行一次连通域检测是行不通的。我的想法就是先做一遍连通域检测进行粗筛，筛选出所有候选框。接着再遍历所有的候选框，首先判断是否是噪声，如果是的话直接删去该候选框；接着再判断是否发生粘连，如果发生粘连，则需要进一步用到字符分割。

对于噪声框的删除，首先我们可以引入三个参数Min\_Area, Min\_Density, MAX\_Aspect，分别表示面积、像素密度以及宽高比。对于面积过小、像素密度过小、宽高比异常的候选框我们直接删除。这样不仅删除了孤立白点，还删除了细碎线段。当同一个字符的外框 A 完全包含另一更小的框 B（B 仅占 A 面积的 < 10 %）时，视 B 为冗余噪声框，予以删除。这样既避免“一个字符两个框”的重复，也可去掉字符轮廓内部的小残片。过滤掉噪声过后的连通域检测结果如**图15**所示

图片包含 图示

AI 生成的内容可能不正确。

图15过滤掉噪声框之后的连通域检测结果

**3.5实现字符分割**

字符分割技术是在连通域检测完毕后，对仍然粘连的图像进行再分割，得到独立的字符图像的技术。对每个连通域，先计算它的包围盒宽度,高度以及宽高比。如果明显大于单字符宽度或者宽高比大于一个阈值，则说明发生了粘连，此时就需要用到字符分割。

**3.5.1投影分割**

投影分割就是将二值化后的文字图像，分别在水平方向（行投影）和垂直方向（列投影）上统计前景像素（即手写数字像素）的累加和，得到一维的投影曲线（也叫投影直方图）

行投影：

列投影：

:位置的像素值，前景对应1，背景对应0.

首先在行投影曲线中查找低谷(接近0的区间),作为不同行之间的边界。对切出的每一行图像，计算其列投影曲线，再在上寻找谷值，可设定一个投影阈值T，当或才认为是切分点。

单纯利用投影分割的结果如**图16**所示。

*图片包含 图形用户界面

AI 生成的内容可能不正确。*

图16采用投影分割的分割结果

可以发现，仅靠列投影，遇到斜写、弧线重叠、字符靠得非常近的情况就会无法正确分割甚至不分割。这是因为投影算法依赖列上必须存在明显的0 像素凹谷才能找到切线；一旦笔画互相穿插或凹谷过窄，就检测不到谷值。

**3.5.2K-means分割**

K-means 不依赖投影谷，而是直接对连通域前景像素的 x 坐标做 1-D 聚类，通过数据驱动的方式来寻找切割边界。不管笔画多斜、多弯，只要两枚字符主要分布在不同列区间，X 坐标聚类就能把它们分成两簇。这也保证了K-means算法不会误切宽高比异常的单字符候选框。我采用的是基于像素坐标的K-means聚类。

首先将图像中所有前景像素的二维坐标作为样本点，当不同字符区域大小差异较大时，直接用聚类容易受尺寸影响，通常将坐标标准化到[0,1]:

**(:**当前连通域中，第i个前景像素点再原图中的坐标。

**:**连通域前景像素在各坐标维度中的最小和最大值。

**:**归一化后，像素点(在区域内的坐标，用于消除不同字符尺寸或分辨率差异对聚类的影响。

归一化的本质是消除尺度差异。假设一个粘连字符的候选框宽度是200px,高度是50px。若直接使用原始坐标做K-means，簇中心的初始值、距离计算等等都会被绝对像素数主导。归一化以后，所有x都被线性映射到[0,1]，所有y都被映射到[0,1]，避免了尺度差异所造成的影响。且归一化不会使得笔画更加粘连，它只是对整个候选框里的坐标按同一比例缩放到[0,1]。这样，任何两个像素在横向或者纵向的相对距离比例并不会改变。

然后进行K-means聚类。对每个样本计算到每个中心的欧式距离：

这样就可以把分配到最近的簇。对每个簇j内的点集，重新计算中心：

当中心距离低于阈值或达到最大迭代次数时候算法停止。如果某个簇内像素点过少（小于像素数阈值），可认为是噪声簇，直接丢弃；如果出现空簇（某次迭代后某个中心无人分配），可用最大 SSE 簇内的点重新初始化该中心，或干脆删掉该簇并减小 k。

若两个簇的外包框高度、宽度、位置非常接近（IoU > 0.5），则合并为同一字符，避免过分拆分；若一个簇的外包框宽高比仍远超单字符比例阈值，可对该簇再递归做一次 K-means细分。

在实际手写字符分割中，仅凭投影曲线出现双峰且谷值足够深来判断是否需要切分往往是不可靠的。当笔画交叠、笔画粗细不均或噪声干扰时，投影峰–谷差可能并不能反映真正的字符分界。为此，我们首先对每个候选框做连通域预筛：如果其面积和宽度与单字符统计特征接近，就认为它是一个整体字符，不做分割；否则，再对该连通域做纵向投影，计算峰–谷差与谷宽度，只有当峰–谷差超过单字符笔画高度的一定比例且谷宽度占候选框宽度的比例足够大时，才在该谷值列处做一次投影分割并验证左右子框面积与宽度都符合单字符范围；若投影预判不满足条件，就不盲目分割，而是先尝试对连通域轮廓做凹陷点检测，如果能在轮廓上找到代表两字符粘连处的显著凹陷，则以凹陷位置为切分线并再次验证子框；如果轮廓凹陷也无法提供可靠边界，则对连通域做骨架细化，寻找骨架分支处的断点，将该断点所在位置作为临时切割线并进行验证；当轮廓与骨架都无法确定分界时，最后采用基于前景像素横坐标的 1-D K-means 聚类，将像素分成两簇，以两簇质心中点作为切割列并验证子框是否满足单字符特征，若仍不符合，则最终将该连通域视为一个整体字符。这样，只有在投影、轮廓、骨架或聚类任一方法都能提供符合单字符统计特征的切分结果时，才真正把候选框拆分为两个字符，否则保留为单字符，保证了在峰–谷假设失效时依然具有鲁棒性。

在投影、轮廓、骨架方法均失效后，采取K-means分割，结果如**图17**所示。

图片包含 图示

AI 生成的内容可能不正确。

图17采用K-means分割的分割结果

可以发现，原来粘连的字符都被分割出来了，充分体现了K-means不依赖于谷值的特征。

**3.6实现字符规范化裁剪**

Otsu阈值法(大津阈值法)就是在灰度图像中，自动寻找一个全局阈值，将图像分成前景和背景，并使两类在灰度空间分离度最大。

假设灰度级集合为(通常L=256),我们对直方图进行归一化处理：

其中为灰度值的像素数，为总像素数，为每个灰度值出现的概率。对于任意阈值，将像素分成两类：

背景类：灰度

前景类：灰度

因此类别概率为：

类别概率为：

类内均值为：

类内均值为：

整幅图像的总均值为：

大津阈值法的核心就是最大化类间方差,该值衡量两类均值之间的可分离性：

当类间方差最大的时候，说明背景与前景在灰度上区分最明显。于是大津法计算：

最终利用阈值分割图像即可。

在单字符 ROI 被检测出来后，本系统采用“一裁、二补、三放、四阈、五缩”的流水线将其规范化为纯黑白小图。首先，保留 ROI 内面积最大的连通域并据此裁剪出最小外接框，然后以字符重心为基准做一次局部几何矫正，使笔画竖直、横平；随后对裁剪图执行闭运算和轻度膨胀，一方面填平 1–2 px 微孔，另一方面把极细笔画整体加粗 1 px，保证骨架完整。接着将处理后的字符放入正方形画布并放大 6 倍，再以 3 × 3 高斯滤波软化锯齿。该放大-平滑过程会在轮廓周围产生大量 60、120 等灰阶像素，此时引入 Otsu 全局阈值。它依据当前灰度直方图自动选取最佳阈值，一刀把化为 0、化为 255，从而将所有灰阶重新压制成纯黑白、恢复真实笔画粗细，并使不同字符图的灰度分布高度一致。最后，以最近邻插值将去灰后的图缩放到 28 × 28 像素并居中输出。至此完成字符规范化裁剪，为 ViT 分类器提供了姿态端正、对比度统一、尺寸一致的标准输入。

**3.7Vision Transformer**

传统卷积神经网络（CNN）在视觉任务中取得了巨大成功，但其局部感受野与平移等变性的设计，也决定了它在捕捉长距离依赖、全局上下文方面存在一定局限。Transformer 在自然语言处理（NLP）领域通过自注意力机制（Self-Attention）成功建模序列中任意位置的关系，尤其擅长捕捉长程依赖。Vision Transformer (ViT)首次将纯 Transformer 架构直接迁移到视觉分类任务，展示了在大规模预训练下优于或可匹敌 CNN 的潜力。ViT网络框架图如**图18**所示。

图示

AI 生成的内容可能不正确。

图18ViT网络框架图

**3.7.1Patch切分与线性投影**

第一部分比较简单，就是将输入的原图像切分为个不重叠的正方形小块(Patch),每块的大小为。然后将每个patch展平成向量，并通过同一个线性映射(全连接层)投影到维度D：

输入图像张量:表示分别表示图像的高度和宽度，本系统中输入的手写字符图像大小为，因此。表示通道数，由于本系统中输入手写字符图像是灰度图，故。

在本系统中，Patch大小，故每个Patch为，切分后Patch的数量一共为7x7=49个。

:表示第个Patch的像素向量。整个Patch是一个形状为的三维张量。先做展平，即将的立方块拉长成一个长度为的一维向量。

:线性映射矩阵，是一个可学习的投影矩阵，将每个展平后的Patch投影到一个维空间。本系统中，通过不同超参组合并进行交叉验证，得到最佳维度为维。说明，，只要进行简单的矩阵乘法和加法就可以完成线性映射了。最后得到映射完的结果序列。但是在代码实现中我们往往用一个二维卷积来代替这个线性映射矩阵，只需要将输入维度设置为1，输出维度设置为64，然后卷积核大小为 ，步长为4，padding填充设置成0，即可等效实现线性映射。

**3.7.2位置编码(Positioning embedding)**

为了保留各patch在原图中的空间位置信息，给每个投影向量加上一个可学习的位置嵌入,同时在序列最前端加上一个分类标记(class token)，其初始值也是可以学习的。最终得到的序列如下：

**分类标记:**这是一个可学习的分类标记向量，形状同样为，在transformer输入序列中它会被放在最前面，用于最终输出全局图像分类的汇聚操作。

**位置编码矩阵**：表示第个token的可学习位置编码矩阵，则表示给token使用。它的目的是让transformer感知到每个Patch在原图中的空间位置信息，因为自注意力机制本身对顺序不敏感。因此为每个投影向量都需要加上相应的位置编码矩阵。因此在本系统中的维度为,其每一个分量都是一个拼接好的序列。

**3.7.3Transformer Encoder**

ViT采用标准的Transformer Encoder堆叠层，每层包含多头自注意力(Multi-Head Self-Attention，MHSA)、前馈网络(Feed-Forward Network，也称MLP)、残差连接加LayerNorm。对第层的输入，其计算为：

**(1)LN(Layer Norm):**层归一化操作，对输入的每一行(对应一个token的64维向量)进行归一化，使得该行均值为0，方差为1，再通过可学习的尺度和做仿射变换。在本系统中，输入为，对第i行进行LN操作：

**(2)MHSA(Multi-Head Self-Attention):**输入。首先先线性映射生成Q、K、V矩阵:

其中是可学习参数，于是。接着我们需要划分需要进行头的划分，在本系统中，通过超参组合学习，得到的最佳头数是个头，每个头的维度是，于是我们将得到的reshape为,对于第个head的。

在划分完头之后，计算单个头的scaled Dot-Product Attention:

其中得到的分数矩阵，为缩放系数，进行归一化操作：

最终输出。拼接所有个head的输出：

再经过线性映射得到最终输出：

输出形状仍然为，把上述MHSA的输出与原投影向量输入进行残差连接，即逐元素相加，得到最终输出，这样能有效缓解网络梯度消失问题。MHSA可以让同一层里不同Patch之间能够互相看见彼此并借鉴彼此的信息。

**(3)前馈网络(MLP):** 在 MHSA 完成了跨token的信息传递之后，给每个 token本身的特征做一次深一点、强非线性的变换。输入。首先先经过一层全连接层将64维投影到中间隐藏层维度，本系统中。然后经过一个GELU激活(后接dropout)：

也可以写成它的近似式：

然后再经过一层全连接层，把隐藏层维度投影到原来的64维，再接一次dropout，这会丢弃一部分特征，使训练时更鲁棒。最后与MHSA同理，需要进行残差连接把MLP的输出与输入相加。在本系统中，总共堆叠L=6层，每层均输出，第层的输出会被用作下一层的输入。

**3.7.4分类网络(MLP Head)**

最后一层编码器输出,取第0个位置，也就是一开始class token的向量,通过一个简单的MLP或全连接层映射到类别数，并做softmax预测：

分类网络首先经过一个全连接层(然后使用tanh激活，使得特征归一，中心对称，有助于后续的线性映射更稳定。之后再做一层全连接层(，得到为归一化的类别分数logits，最后使用softmax将其转换成概率分布，选择概率最大的作为其类别标签。

最终，可以画出基于ViT网络实现的分类器的详细设计流程图，如**图19**所示。

图示, 示意图

AI 生成的内容可能不正确。

图19ViT分类器设计流程图

**4.实验结果分析**

**4.1ViT分类器**

在训练出最终的ViT分类器前，我们首先需要确定超参。MNIST的训练数据是60000，我将训练数据进一步划分为训练集和验证集，比例为8:2，即再分后的训练集为48000，验证集为12000。我设置batchsize为128，那么一个周期的迭代次数为48000/128=375。对于超参的每一种组合我都跑10个周期，并记录每个周期的training loss以及validation accuracy。然后等每一组超参跑完，更新最佳的超参组合。等最佳的超参组合选定，再开始正式的训练，得到最终的ViT模型。训练过程图如**图20**所示。

文本

AI 生成的内容可能不正确。

图20ViT模型训练过程图

生成最终的ViT模型后，打印输出training loss以及validation accuracy的曲线图，如**图21**所示。

图表, 折线图, 散点图

AI 生成的内容可能不正确。

图21最终模型的training loss以及validation accuracy

最终得到ViT模型在测试集上的准确率达到了99.4%,效果非常好。

**4.2提取单字符输入集**

在训练完ViT模型后，我们需要将待识别的手写数字输入图转换成单个字符大小为28x28的字符输入集合，用于预测训练。本系统的手写数字输入图如**图22**所示

在桌子上

AI 生成的内容可能不正确。

图22手写数字输入图

首先先调用PaddleOCR进行字符所在行检测，检测结果如**图23**所示。

白板上写着字

AI 生成的内容可能不正确。

图23PaddleOCR检测字符所在行结果

通过自主筛选出检测到的非字符所在行的区域，也就是红色的夹子，得到定位到字符所在行的输入图像如**图24**所示。

白板上写着字

AI 生成的内容可能不正确。

图24定位到字符所在行的输入图

紧接着，用自适应阈值法与形态学操作，得到二值化掩模图，前景为白色数字，背景为黑色，二值掩膜结果如**图25**所示。

文本

AI 生成的内容可能不正确。

图25二值掩膜结果

然后再进行PCA几何矫正得到经过矫正过后的二值掩膜结果图，如**图26**所示。

黑板上的字

AI 生成的内容可能不正确。

图26经PCA几何矫正后的二值掩膜结果

最后再运用Otsu、连通域检测、字符分割等多种方法实现单个字符的检测与分割，检测分割结果如**图27**所示。

形状

AI 生成的内容可能不正确。

图27单个字符的检测分割结果

裁剪到的输入图像样例如**图28**所示。







图28裁剪得到的单个字符的输入集合

**4.3送入分类器进行预测**

最后我们把得到的单字符集合送入训练好的ViT网络进行预测，得到的最终预测结果图如**图29**所示。

图片包含 应用程序

AI 生成的内容可能不正确。

图29ViT分类器预测结果

可以发现，全部识别正确！

**5.实践总结**

在本次课程实践项目中，我基于Vision Transformer（ViT）网络设计并实现了一套完整的手写数字字符图像检测与识别系统。在预处理与字符提取方面，我结合了自适应阈值方法和多阶形态学操作，有效解决了实际环境中光照不均、噪声干扰以及字符粘连的问题。此外，我利用主成分分析（PCA）成功进行了几何矫正，大幅提高了后续字符切分的准确性。通过连通域检测、投影分割和K-means聚类等策略，最终精确提取了规范化的单字符图像，为ViT分类器提供了高质量输入。

在ViT分类器设计方面，我选择了Vision Transformer模型，以其全局自注意力机制很好地捕获了手写字符图像的整体结构特征，弥补了传统CNN在处理长程依赖时的不足。经过在MNIST数据集上48000个样本的训练，分类器最终在验证集上实现了99.4%的准确率。此外，在真实测试场景中，系统实现了完全正确的分类结果，充分体现了预处理和ViT模型结合后的整体优势。

总体而言，本系统在复杂场景下表现出稳定性和较好的通用性，体现了深入的图像预处理与先进模型结合的价值。然而，也存在一些不足之处，主要体现在：字符分割模块和图像预处理的某些阶段，仍然依赖于经验性阈值和主观参数设定，未实现真正意义上的全自动化。此外，尽管ViT模型在GPU环境下推理速度可以满足实时需求，但在CPU上运行效率仍有优化空间。这些问题是未来进一步改进和研究的方向。实践过程中的照片如**图30**所示。

人拿着笔记本电脑

AI 生成的内容可能不正确。人在玩电脑

AI 生成的内容可能不正确。

图30实践过程照片

**参考文献**

[1] A. Dosovitskiy *et al.*, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale,” *arXiv preprint arXiv:2010.11929*, 2020.

[2] M. Liao, Z. Wan, C. Yao, K. Chen, and X. Bai, “Real-time Scene Text Detection with Differentiable Binarization,” *arXiv preprint arXiv:1911.08947*, 2019.