拍照矩阵计算器开发——手写矩阵识别

Matrix in Sight

1st 黄哲昊
上海交通大学
电子信息与电气工程学院自动化系
上海，中国
E-mail：KinghtH@outlook.com

2nd 叶瑜超
上海交通大学
电子信息与电气工程学院自动化系
上海，中国
E-mail：mestyeyc@sjtu.edu.cn

3rd 陈禹衡
上海交通大学
电子信息与电气工程学院自动化系
上海，中国
E-mail：chenyuheng@sjtu.edu.cn

摘要: 手写矩阵的识别不同于简单的单个手写数字识别，由于矩阵维度的不确定性及矩阵内部各元素所呈现的位置和内容信息的多样性，一直以来都是光学字符识别（下文简称 OCR 识别）任务的一大难点。我们将手写矩阵识别分为数字字符位置识别与多位手写数字识别两个阶段，并将两者的结果使用原创的基于 K 近邻的矩阵元素整合方法组合得到数值矩阵结果。并在全部由我们自己生成或收集整理的标准 Latex 矩阵数据集和手写矩阵数据集都取得了稳定、准确的识别效果。

关键词: 目标检测，OCR 识别，时序模型，K 近邻

Abstract: Different from single handwritten digit recognition, the recognition of handwritten matrices has been one of the challenges in optical character recognition all the time because of the uncertainty of dimensions of matrices and the diversity of locations and contents of the numbers in the matrix scripts. We divide the task into two stages of digits and signs position recognition and handwritten multi-digit recognition, which will be combined by graph-based matrix element integration method to get the numerical results. And our algorithm is able to achieve stable and accurate recognition results on both standard Latex matrix data set and handwritten matrix data set which are all generated or collected and then labeled by our own.

Key Words: Object detection, Optical Character Recognition, RCNN, Sequence Model, Graph theory

I. 绪论

A. 应用背景介绍

” 一切问题都可以转化为数学问题，一切数学问题都可以转化为代数问题，而一切代数问题又都可以转化为方程问题。” 这是来自法国数学家笛卡尔的名言，线性方程组的求解作为方程问题中最基础也是最重要部分，能够求解复杂线性方程组的有力工具——线性代数的重要性也不言而喻。线性代数矩阵理论本身作为能够表示空间的线性变换最直接的数学工具，在众多学科领域都有着广泛的应用。而且随着机器学习的迅猛发展，其中涉及大量的向量矩阵运算，使得对于高效的线性代数运算有了更高的要求。

而对于研究人员，假如能够将自己手下推导的矩阵运算公式快速输入到计算机中进行运算，其研究效率必然会大大提高。在此之前就已经有将手写数学公式转换为 Latex 公式或进行计算的应用。而对于照片中目标矩阵运算的识别及计算的应用却非常少见。因此我们希望能够使用机器学习方法，设计并实现一个能够识别图片中需要进行运算的矩阵以及运算操作的应用，并最终得到需要的运算结果。

B. 应用实现流程简介

识别手写矩阵的运算公式这个任务本身属于 OCR 分析识别问题。但不同于简单的单个数字分类，矩阵维度的不确定性以及手写矩阵内部元素的位置和多位手写数字内容所构成的复杂程度，是传统 OCR 识别方法所难以解决的。因此我们尝试使用机器学习方法结合深

度学习模型来解决手写矩阵识别这个问题。但是仍由于手写矩阵本身特征的复杂性，目前常见的神经网络架构难以实现单个模型直接进行手写矩阵数据的训练测试并最后部署的端到端应用开发流程。因此我们将手写矩阵识别任务分为数字符号位置识别（III-A）和多位手写数字识别（III-B）两个阶段，即先找到包含不同数字和运算符号的区域，再对区域中的数字或符号进行识别分类，我们也使用了两个不同的深度学习模型分别处理这两个阶段的任务，并且我们使用了原创的基于 K 近邻的矩阵元素整合（III-C）方式将两个阶段得到的结果合并为可以直接进行运算的数值矩阵。两个识别阶段以及矩阵元素整合方法的详细解释见第三章手写矩阵识别算法（III）部分。

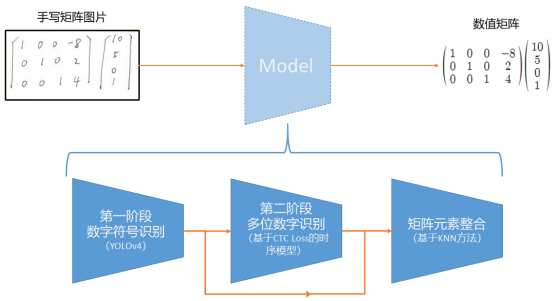

图 1. 应用实现流程图

C. 论文组织结构

全文组织结构如下：

- I-绪论：介绍了手写矩阵识别的应用背景并简单介绍了我们实现手写矩阵识别应用的算法流程；
- II-相关算法概述：首先指出传统 OCR 算法在识别手写矩阵任务上的弊端，再分别介绍用于处理数字符号位置识别与多位手写数字识别任务所使用的两个的深度学习模型——YOLOv4 和基于 CTC Loss 的时序模型；
- III-手写矩阵识别算法：详细介绍数字符号位置识别与多位手写数字识别的算法实现，以及基于 K 近邻的矩阵元素整合方法的具体实现，并解释如何将两个阶段的结果进行拼接得到最终的数值矩阵；
- IV-数据集构建：介绍训练 YOLOv4 模型时所使用的标准 Latex 数据集的生成方式以及数据标注规则，以及如何标注所收集到手写矩阵，同时还会介绍用于训练时序模型的多位数字图片数据集的详细信息。
- V-训练过程与测试结果：展示模型的训练过程具体信息以及在验证数据集上的测试结果，并展示算法实际应用在拍摄图片识别手写矩阵时的效果。
- VI-总结与展望：对手写矩阵识别应用算法进行总结，并对未来工作进行展望。

II. 相关算法概述

A. 传统 OCR 算法的弊端

在深度学习正式引入 OCR 检测任务之前，传统目标检测方法都是先检测和提取文字区域，接着利用直线曲线检测的 Randon 变换与 Hough 变换进行文本校正 [1]，通过投影直方图分割出单行的文本图片，这样就把多行的 OCR 问题转变为单行 OCR 检测问题，然后对单行进行分割字符（segmentation based method），如基于投影直方图极值点作为候选分割点搜索最佳分割位置，对于搜索得到的单个字符，使用灰度处理、二值化、矫正图像后，人工地提取图像特征，最后对特征使用支持向量机或是 K 近邻等算法分类。但是这样的传统方法就有两个难以解决的问题：

1) 文字区域检测以及分割点选择的策略效果差，时间复杂度高，在遇到手写矩阵时可能无法直接对原始图像分割出单行 OCR 进行进一步识别，而且在遇到数字字符粘连时难以进行分割处理；
2) 手工提取的图像特征如颜色、字体以及形状的鲁棒性较差。

B. 目标检测网络 YOLOv4 [2]

为了处理特征复杂的手写矩阵图片，我们引入深度学习目标检测网络 YOLOv4 对矩阵图片进行第一阶段处理。YOLO 将检测任务表述成一个统一的、端到端的单阶段回归问题，并且以只处理一次图片同时得到位置和分类而得名。网络结构包括四部分，YOLOv3 [3] 作为读取头，CSPDarknet53 作为骨干网络，SPP [4] 作为 Neck 的附加模块，PANet [5] 作为 Neck 的特征融合模块。其主要优点在于首先在保证检测准确率的前提下，可以达到实时检测的速度；而且 YOLO 网络对于背景的误检率低，在检测过程中，能够直接识别整张图像的整体信息；并且检测的泛化性能较好。我们所使用的 YOLOv4 在目标检测任务领域实现了最优的精度速度平衡（图2）。

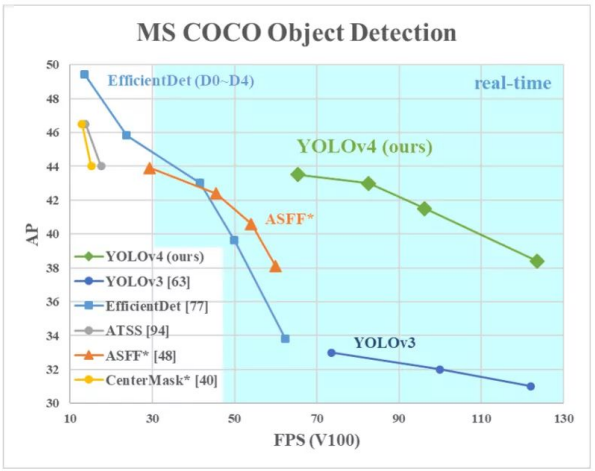

图 2. YOLOv4 及 SOTA 算法在 COCO 数据集上的实验结果

YOLOv4 网络最终会输出多达 10647 个锚框，但通常图片中并没有这么多个目标需要进行检测，其首先使用阈值过滤掉置信度低于设定值的锚框，然后使用非极大值抑制（NMS）[6] 的方法解决同一个物体上出现多个锚框的问题，即当多个锚框重合度较高时只选择置信度最高的框。其中重合度 IoU 的计算如下图3所示。手写矩阵通常不会出现两个高度重的检测目标，但在实际第一阶段检测结果中，经过非极大值抑制后仍会出现多个框检测同一个数字或运算符号的现象，因此我们在第一阶段中对 YOLOv4 原本的非极大值抑制进行了一些改进，使其检测结果对于手写矩阵的检测更加稳定和准确。

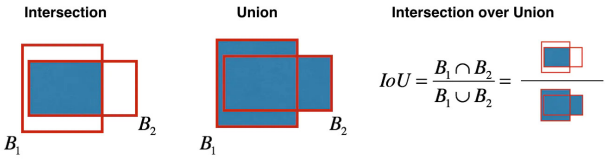

图 3. IoU 计算方式

C. 基于 CTC Loss 的时序模型 [7] [8]

矩阵图片经过 YOLOv4 处理将得到矩阵内各个数字的位置信息并以此获得这些多位数字图片，常用的处理算法有 KNN [9]、SVM [10] 分类与神经网络，我们使用基于 CTC loss 的时序神经网络模型 [8] 对多位数字图片进行识别。

CTC(*Connectionist Temporal Classification*) 是一种用于序列建模的工具，其核心是定义了一个特殊的目标函数，使输出与标签对齐。在本阶段中使用的模型为 CNN+LSTM [11] 构建的网络结构，但对于一个简单的网络模型，输出位数是固定的，多位数字的标签却长短有变，其输出和标签长度通常是不对等的，因此需要设计损失函数，利用 CTC 算法将输出与标签对齐。

图 4. CTC 标签对齐示例

以图4为例，其中网络输出为 14 位，但标签位数为 4 位，CTC 算法利用特定解码函数（称为 β 函数）对网络输出进行转换，得到四位标签。对这个 4 位的标签值同时存在多个对应正确的网络输出，算法的基本思想就是将模型输出的定义为所有可能对应的标签序列上的概率分布，因此优化目标就是最大化得到正确标签的概率。

根据以上想法，其定义，在给定模型网络输入为 x 时，输出的目标序列 l 的条件概率为

$$p(l|x) = \sum_{\pi \in \beta^{-1}(l)} p(\pi|x) \tag{1}$$

其中 β^{-1} 即为 CTC 加密解码算法，$\beta^{-1}(l)$ 表示所有可能被解码为目标标签的网络输出序列。同时，因为 CTC 假设输出概率条件独立，因此有

$$p(\pi|x) = p(\pi|y = N_w(x)) = \prod_{t=1}^{T} y_{\pi_t}^t \tag{2}$$

其中 $y_{\pi_t}^t$ 为在输出序列为 π 时每一输出位相应的概率。

结合公式1，对所有可能的输出序列概率求和即可得到优化函数。但需要注意的是，对应路径数目计算公式为 C_{T-1}^n(n 为数字位数)，量级约为 $(T-1)^n$，计算量过大，因此 CTC 借用了 HMM 模型中的向前向后算法进行动态规划的计算 [12]，极大减少计算量。得到优化函数后即可对模型训练，进行多位数字图片的识别。

III. 手写矩阵识别算法

我们使用第一阶段先对矩阵数字元素以及运算符号位置进行识别，第二阶段对识别得到的文字区域进行

序列识别，得到其中的数值，最后用基于 K 近邻的矩阵元素拼接方法将两个阶段得到的信息整合在一起得到数值矩阵结果，实现了稳定的从标准公式截图和自然场景图片到正确可计算输出的手写矩阵识别算法。

A. 第一阶段数字符号位置识别

将原始输入图片输入到 YOLOv4 目标检测网络后可以得到一个包含锚框信息的输出（图5）。

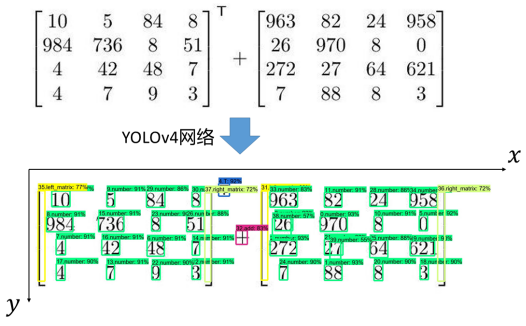

图 5. 第一阶段数字符号位置识别输出结果示例

1) 输出锚框信息： 输出中会包含所有候选锚框的信息列表 $[label, x_{min}, y_{min}, x_{max}, y_{max}, score]$。下面会分别对这些信息进行解释：

- *label*:
  表示锚框区域内容的类别，我们在第一阶段识别中，将类别分为 $\{number$(数字)$, add$(加)$, minus$(减)$, multi$(矩阵乘法)$, T$(矩阵转置)$\}$，其中数字类识别得到的区域将作为输入在第二阶段输出其真实数值；后四个类别的内容直接被识别得到并应用在元素整合过程中，
- $x_{min}, y_{min}, x_{max}, ymax$:
  这四个值分别表示锚框在图片左上角点和右下角点在图片坐标系下的坐标值，直接表示了锚框的区域位置，并可以通过简单计算得到其面积及中心点坐标，这些信息不仅会在我们改进的非极大值抑制算法中用于计算 IoU，而且将在矩阵元素整合过程中发挥重要作用。
- *score*:
  最后一个锚框的信息是置信度，当置信度越高时，代表类别概率和位置信息概率越高，在 YOLOv4 原本的输出处理算法中按置信度对锚框进行阈值过滤和非极大值抑制保留来得到最终的锚框，我们在我们改进之后的非极大值抑制算法中也将置信度作为主要参考依据来对锚框进行去除和保留。

2) 改进非极大值抑制： 由于使用原始 YOLO 的非极大值抑制算法后仍会在部分手写矩阵识别图像中产生重叠框，因此我们对其进行了改进，根据我们手写矩阵识别任务的实际特性，引入候选框分类信息来对不同类别的候选框做不同的处理：

- 对于数字图像区域我们将候选框面积最大的进行保留，这是因为我们需要保证所有数字都被包含在候选框内作为第二阶段多位数字识别的输入，而且即便包含了一些空白区域或是其他数字的干扰，我们也考虑到第二阶段识别时使用不同样本来减少干扰带来的影响；
- 当候选框类别为运算符号时，由于 YOLO 网络直接给出了运算符号对应的类别，因此对于候选框的信息我们事实上并不关心，因此我们取重叠候选框中置信度最高的进行保留，并经过验证得到了非常高的准确率。

同时在计算重叠时，我们采用了直接判断两个候选框的中心是否在相互的候选框内来作为两个候选框重叠的判断，由于实际应用环境中手写矩阵中元素重叠较少，因此使用简单的中心重叠判断能够代替 IoU 计算来快速准确地判断两个候选框是否重叠并进一步处理。

B. 第二阶段多位手写数字识别

经过 YOLO 网络得到所有标注框位置信息后，利用这些信息进行数字图片的截取、处理并识别。

1) 图片截取过程： 利用标注框信息，并判断此标注框是否属于数字，若属于数字则以边缘信息将其在图片中截取出来。

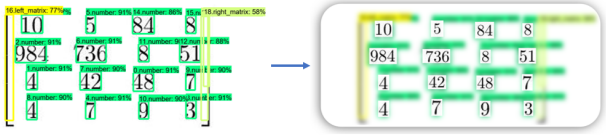

图 6. 图片截取过程

对于截取出的图片，形状不一，需要对其进行大小的统一以及一定的预处理，满足网络模型的输入维度要求。

电子版矩阵数字比较标准，且大部分均为黑底白字，因此不需要进行二值化处理，直接对其进行图片形

状重塑即可。考虑到直接进行水平、垂直拉伸将改变数字相对结构，对识别造成影响。我们先将截取出图片等比例放大至与某一维度与要求维度相当，并用白色像素点将其余位置填充，得到一张处理符合要求的图片，如图7所示。

图 7. 电子版数字预处理

但对手写矩阵，由于拍摄环境不同，像素点信息会有所不同，对图片要进行一定预处理，使其成为较标准的图片便于识别。如图8所示

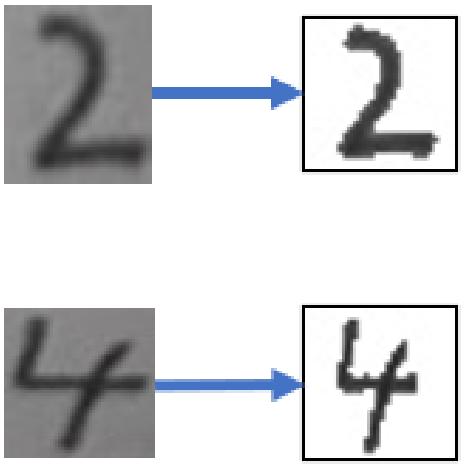

图 8. 手写版数字预处理

预处理过程主要包括两步：

1) 灰度化。首先将读取到的 RGB 图片信息转化为灰度信息，便于二值化处理。
2) 二值化。设定阈值为所有像素点灰度值均值减去 10，当像素点灰度值小于阈值时，认定其为数字的有效黑色像素点；而灰度值大于阈值时，认为其为背景信息，相应设定为灰度最大值 255。

如此得到一张预处理后的图像。并进行类似于电子版形状大小变换的过程，使其满足网络输入要求。

2) 识别过程： 利用基于 CTCloss 的时序网络模型对图片直接进行识别，返回识别结果。同时我们也搭建了识别 MNIST 数据集的网络模型，进行简单的单个数字识别以验证算法流程的正确性。

C. 基于 K 近邻的矩阵元素整合

当得到第一阶段 YOLO 网络输出的经过改进的非极大值抑制的锚框信息以及第二阶段多位手写数字识别得到的每个数字候选区域内的识别结果后，就需要将每个数字排版到矩阵中对应的位置，和运算符一起组成最终的运算式。但是由于事先并不知道单个矩阵的维度信息，同时由于单个手写矩阵内每个数字位置并不会一一对齐，而是可能出现一些部分重叠或者倾斜，这就使端到端的深度学习方法和基于传统 OCR 检测的方法在手写矩阵识别任务上难以得到有效的应用。我们所使用基于 K 近邻的算法在考虑到单个手写矩阵图像本身具有一定的几何特征，对于其中每个元素只考虑最邻近的 4 个上下左右位置的元素，组成关于手写矩阵图像的无向连通图。然后就能够根据矩阵和图的特性从对应于矩阵中第一行第一列的元素开始遍历得到整个数值矩阵以及其行数和列数，完成矩阵元素整合。

1) 根据运算符分割图片： 第一阶段输出所有锚框信息后只有独立的单个锚框的类别及其在图片坐标系下的位置信息，首先我们根据其中运算符的位置，对将各个锚框划分到对应的矩阵元素集合内，其流程如下：

1) 找到图中锚框类别为运算符的位置信息；
2) 假如图中没有运算符（不包括转置），则默认为取行列式，即将所有锚框纳入到一个矩阵的元素集合中；
3) 假如图中有一个运算符（不包括转置），则以运算符 x 坐标位置将图片划分为两个区域，两个区域各自形成一个矩阵元素集合，各个区域中假如出现转置符号则表明在运算时该矩阵需要进行转置操作；
4) 假如图片中检测出超过一个运算符（不包括转置），则会提示错误信息。

2) 锚框信息投影到欧氏空间坐标系： 由于需要进行 K 近邻操作，因此需要根据锚框信息计算得到每个锚框的中心点，并将他们投影到欧氏空间坐标系下，方便之后进行中心点间距离的计算（图9）。

图 9. 投影到欧式空间

3) 基于 K 近邻的矩阵元素无向图构建: 原本的 K 近邻算法相当于以目标点为圆心扩大圆的半径计算得到与圆心最近的 K 个点，但是我们考虑到每个矩阵元素事实上只需要知道其上下左右相邻位置元素的信息就可以通过关联信息扩展得到整个矩阵，因此我们对于每个元素划分了四个区域分别计算与中心元素最近的四个元素（图10），即 K 近邻中 K 取 1 的情况。假如划分的区域中没有元素，那就表明该元素可能位于边界或是顶点。假如中心元素的坐标为 (x_0, y_0)，遍历其他元素 (x_i, y_i)，并得到锚框中长或宽中的最大值 l_{max}，可以通过如下判断方式将其他元素划分到中心元素相邻的四个区域中进一步进行欧式距离的判断:

- $|y_i - y_0| < \frac{l_{max}}{2}$ and $x_0 > x_i$: 中心元素左边区域;
- $|y_i - y_0| < \frac{l_{max}}{2}$ and $x_0 < x_i$: 中心元素右边区域;
- $|x_i - x_0| < \frac{l_{max}}{2}$ and $y_0 > y_i$: 中心元素下边区域;
- $|x_i - x_0| < \frac{l_{max}}{2}$ and $y_0 < y_i$: 中心元素上边区域;

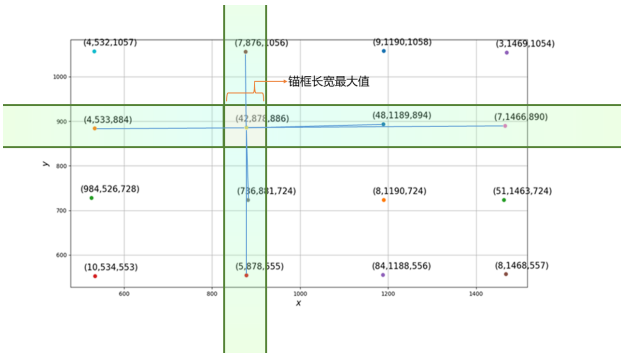

图 10. 在四个区域中分别基于 K 近邻判断是否相邻（K=1）

在得到所有矩阵元素组成的只包含相邻元素连接关系的图后（图11），首先就是寻找矩阵中第一行第一列的元素，由于图片坐标系与欧式空间坐标系是成一个上下反转的关系，因此我们实际上是寻找图中下边区域和左边区域没有相邻元素的节点，就对应了矩阵中最起始的元素，然后我们就可以通过遍历该元素右边及上边元素所构成的图来计算最终矩阵的列数和行数，并通过遍历整个图来得到整个数值矩阵了。

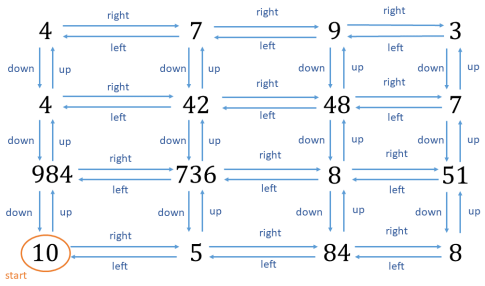

图 11. 只包含相邻元素连接关系的图

IV. 数据集构建

A. *Latex* 矩阵图片数据集生成及标注规则

因为手写数据集相对复杂且较难收集及标注，我们首先对电子版矩阵进行识别处理。为使生成图片更符合现实中电子版矩阵样式，生成电子版数据集使用到 pylatex 包，其支持在 python 环境下进行 latex 式编辑，从而生成 latex 版矩阵图片。但因为 pylatex 只支持生成 pdf 类文件，因此还需要进行 pdf 转图片并截取相应位置矩阵，获得所需数据集。整体生成流程如图12所示

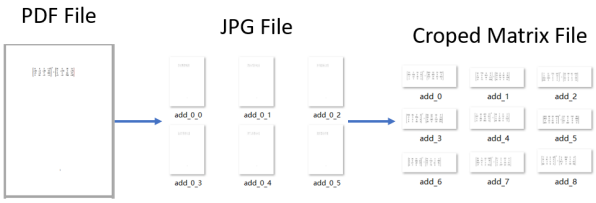

图 12. 电子版矩阵生成流程

基于此流程，共生成电子矩阵图片 1680 张，共包括以下几种形式变化:

- 三种运算方式（加、减、乘）
- 四种字体
- 转置符号
- 不同形状大小

其中所有数字随机以及随机截取位置，以此保证数据集的多样性。

不同于使用 PIL 库直接生成图片，为了生成更符合实际使用的 latex 形矩阵，我们无法得到相应多位数字的位置信息。所以在之后的工作中需要对所有矩阵信息进行标注。为此，我们利用 opencv 库对图片进行一定检测设计出辅助人工标注的程序，极大提高标注效率。

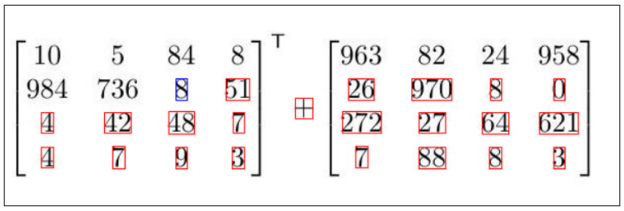

图 13. 矩阵标注程序显示

首先，对图片进行一定膨胀预处理，使得多位数字轮廓粘合可被视为整体。利用 opencv 库中的轮廓检测函数对矩阵图片进行检测，此时可以得到多个轮廓，其中包括矩阵内的多位数字、运算符号、转置符号以及左右矩阵框符号。检测函数无法做到不同符号的区分，因此需要人工处理。在得到轮廓信息后，标注程序将逐个显示轮廓，用户通过按下相应按键即可完成对此轮廓边框的信息标注，操作显示如图13所示。其中需要标注的信息及对应按键如下所示

- number : n
- left_matrix : l
- right_matrix : o
- add : a
- minus : m
- multi: t
- T :k

相比于使用直接对图片进行标注处理，我们设计的辅助标注程序极大简化了用户所需操作。

B. 手写矩阵图片数据集收集及标注规则

1) 数据集收集：
   收集手写矩阵图片不同于生成 latex 矩阵图片，latex 生成的矩阵格式较为规整，而考虑到不同研究人员写字风格的差异，手写矩阵图片的来源需要多样化，即需要考虑到不同的字迹，才能使我们的数据集更加具有代表性，提高手写矩阵识别的泛化性。大一《线性代数》课程中使用的作业本恰好符合我们的数据集要求，在初学矩阵时，是使用纯数字矩阵最多的时候，同时不同学生字迹也不尽相同，满足了数据集中的数据来源丰富的要求。因此，我们在征询学生本人同意的情况下拍照获取的线性代数作业本上的矩阵数据，同时将得到的含有矩阵的图片进行切割裁剪，使得每一张图片尽量只包含一个或者两个矩阵，限于时间原因，我们最终收集到 700 多张含有矩阵的照片，用于网络训练。
2) 手写数据集标注：
   为了进行手写数据集的标注，我们首先使用了 labelImg 工具（见 https://github.com/tzutalin/labelImg），该工具需要预定义需要标记的标签，然后在图片中标若干框，同时给每个框选择一个标签。每张图片的标签信息以 xml 文件的形式进行存储。预定义的类别有 number（数字类），left_matrix(矩阵左边框)，right_matrix(矩阵右边框)，*add*(加运算符)，minus（减运算符），multi（乘运算符），*T*(转置运算符)。标注示例如图14。

图 14. 手写矩阵标注示例

C. 多位数字图片数据集

在图片处理流程最后将对多位数字图片进行识别，因此需要生成相应数据集对基于 CTCloss 的网络模型进行训练。多位数字图片分为两种，一种为电子版数字图片，另一个则是基于 MNIST 数据集生成的多位手写数字图片。

1) 电子版多位数字的生成生成基于生成验证码包 capycha，其功能是给定多位数字，随机生成相应验证码图片。CAPYCHa 包的验证码图片生成原理是对原数字图片进行拉伸、旋转等操作并在背景加上噪声，因此我们对其中 image.py 文件进行

修改，以此生成随机字体以及随机大小的数字图片，如图15所示。

图 15. 电子数字图片生成（左边为原验证码图片，右边为生成图片）

2) 多位手写数字图片的生成手写图片生成基于 MNIST 数据集，主要思想是对给定多位数字，在 MNIST 数据集中随机取相应标签的图片，并利用 OPENCV 库将其按照顺序并给定一定随机性贴到相应的背景上，以此生成一个单位数字为 MNIST 数据集图片组成的多位数字图片。同时，因为 MNIST 数据集图片为黑底白字，因此对图片进行处理使其变为黑底白字，便于识别流程的统一。如图16所示

图 16. 手写数字图片生成（左边为 MNIST 数字合成图片，右边为处理后白底黑字数字图片）

V. 训练过程与测试结果

A. *YOLOv4* 在电子版矩阵数据集上训练过程

1) 训练相关设定:

- 电子版矩阵数据集标注框分为 7 类，分别为 *number*（数字）,$left_matrix$（矩阵左括号）,$right_matrix$（矩阵右括号）,*add*（加法）,*minus*（减法）,*multi*（乘法）,*T*（转置）;
- anchor 预选框大小和个数与原 YOLOv4 设定相同;
- 训练集包含 1646 张图片，测试集包含 33 张图片;
- 训练 20 轮。

2) 训练结果: 训练后 YOLOv4 最高在电子版矩阵测试集上达到平均 mAP 为93.3%，其中在数字类识别上最高达到98.21%mAP。在测试集上经过我们改进的非极大值抑制所有第一阶段输出的锚框包含数字成功率达到100%。

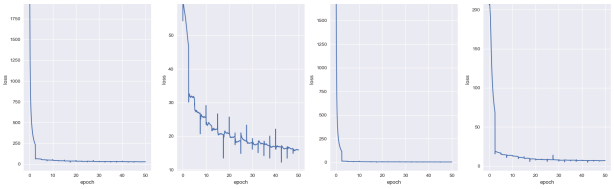

图 17. 电子版矩阵数据集 YOLOv4 训练 Loss 曲线下降图

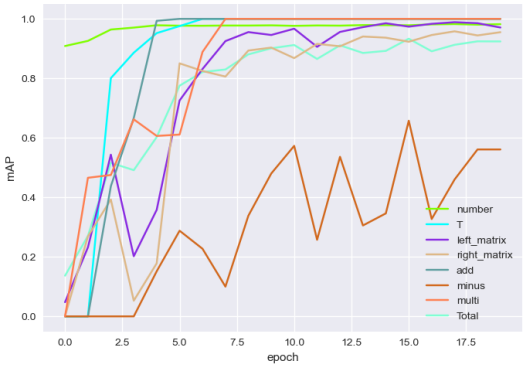

图 18. 电子版矩阵数据集 YOLOv4 在测试集上 mAP 变化曲线

B. *YOLOv4* 在手写矩阵数据集上训练过程

1) 训练相关设定:

- 使用电子版矩阵第一阶段识别 YOLOv4 网络作为预训练模型继续训练;
- 手写矩阵数据集标注框分为 2 类，分别为 *number*（数字）,*T*（转置）;
- anchor 预选框大小和个数与原 YOLOv4 设定相同;
- 训练集包含 768 张图片，测试集包含 89 张图片;
- 训练 50 轮。

2) 训练结果: 训练后 YOLOv4 最高在手写矩阵测试集上达到平均 mAP 为92.1%，其中在数字类识别上最高达到84.76%mAP。在测试集上经过我们改进的非极大值抑制所有第一阶段输出的锚框包含数字成功率达到98.44%。

C. 基于 *CTC Loss* 的时序模型在电子版多位数字数据集上训练过程

1) 训练相关设定:

- 使用第一阶段截取出多位数字图片作为输入（全部靠左排版）;
- 直接输出对应数字;
- 训练集包含 6027 张图片，测试集包含 1000 张图片;

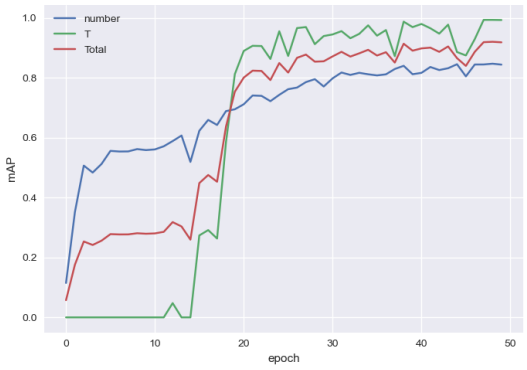

图 19. 手写矩阵数据集 YOLOv4 训练 Loss 曲线下降图

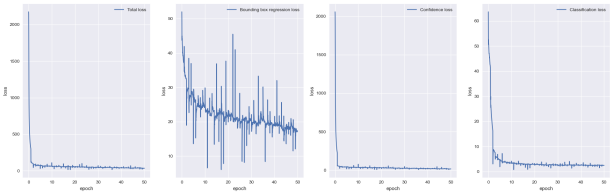

图 20. 手写矩阵数据集 YOLOv4 在测试集上 mAP 变化曲线

- 训练 4 轮。

2) 训练结果: 使用时序模型识别电子版的多位手写数字时正确率较高，在训练集和测试集上都可以达到100%的正确率。

D. 测试结果

完成 YOLOv4 模型与多位数字识别模型训练后，即可进行完整的矩阵识别效果测试。测试精度确定方式为，随机挑选测试矩阵图片并输入模型，观察结果是否符合人眼判定结果，计算识别正确率。

对于电子版矩阵，模型成功识别图片内矩阵及运算符从而完成运算的正确率达到 89%。

对于手写矩阵，模型测试的正确率达到 80%。

VI. 总结与展望

为了解决手写矩阵图片识别问题，我们提出了流程化的二阶段矩阵识别方法以及通过基于 K 近邻方法整合矩阵元素，在电子版矩阵运算图片和手写矩阵运算图片上都取得了稳定且准确的识别效果，并且我们所开发的电子版矩阵运算公式生成方法可以实现大批量的样本生成。但是由于手写版矩阵图片数据集的收集更为困难且需要耗费大量时间，因此我们最终只在手写矩阵第二阶段使用了识别一位手写数字的网络模型进行识别，未来我们考虑不断收集更多背景和样式更加丰富的手写矩阵样本，使得可以在第二阶段也能够对多位手写数字进行识别。并且由于我们手写矩阵识别框架的完整性，今后可以非常方便的对其中任意一个识别模块进行升级，如由于我们大多数识别都是离线进行而不需要追求实时性，因此我们将 YOLO 网络替换为虽然识别速度更慢但准确度更高的 EfficientDet 来实现，并且在今后我们考虑将第二阶段识别网络换成可以识别更多种类数字如支持小数分数识别，并不断优化整个识别过程来实现更加稳定准确的手写矩阵计算照片识别应用。

参考文献

[1] 周冠玮, 平西建, and 程娟, “基于改进 hough 变换的文本图像倾斜校正方法,” Ph.D. dissertation, 2007.

[2] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, “Yolov4: Optimal speed and accuracy of object detection,” 2020.

[3] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *arXiv preprint arXiv:1804.02767*, 2018.

[4] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[5] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, “Path aggregation network for instance segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8759–8768.

[6] A. Neubeck and L. Van Gool, “Efficient non-maximum suppression,” in *18th International Conference on Pattern Recognition (ICPR’06)*, vol. 3. IEEE, 2006, pp. 850–855.

[7] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks,” in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 369–376.

[8] K. Kawakami, “Supervised sequence labelling with recurrent neural networks,” *Ph. D. thesis*, 2008.

[9] S. A. Chaudhari and R. M. Gulati, “An ocr for separation and identification of mixed english—gujarati digits using knn classifier,” in *2013 International Conference on Intelligent Systems and Signal Processing (ISSP)*. IEEE, 2013, pp. 190–193.

[10] F. Bouchareb, R. Hamdi, and M. Bedda, “Handwritten arabic character recognition based on svm classifier,” in *2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications*. IEEE, 2008, pp. 1–4.

[11] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] L. R. Rabiner, “A tutorial on hidden markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

VII. 成员主要工作内容及贡献

- **黄哲昊：**整体识别流程设计，网络结构实现及训练，矩阵元素整合算法设计，代码整合
- **叶瑜超：**latex 电子矩阵数据集生成，验证码图片数据集生成，Multi-MNIST 数据集生成，图片预处理
- **陈禹衡：**手写矩阵数据集收集、数据标签标注

VIII. 附录

A. *YOLO4*

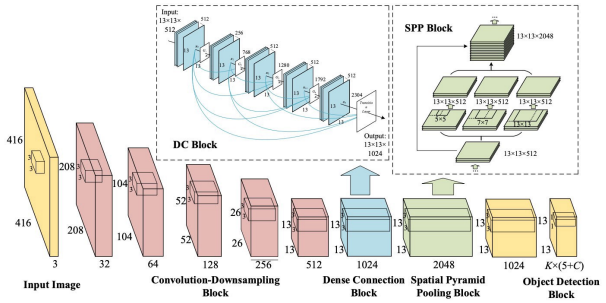

图 21. YOLOv4 网络结构

实现源代码参考https://github.com/argusswift/YOLOv4-pytorch

B. 标签标注软件 *labelImg*

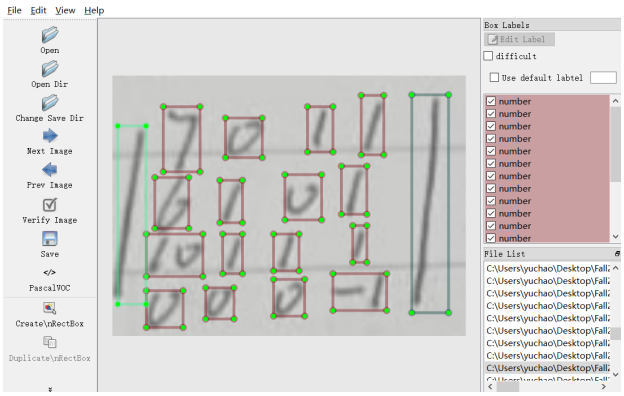

图 22. 标注界面示意图

软件代码及下载参考https://github.com/tzutalin/labelImg

C. 基于 *CTC loss* 的时序网络模型

CTC 网络结构如图23, 实现源代码参考https://github.com/ypwhs/captcha_break/blob/master/ctc_pytorch.ipynbctc

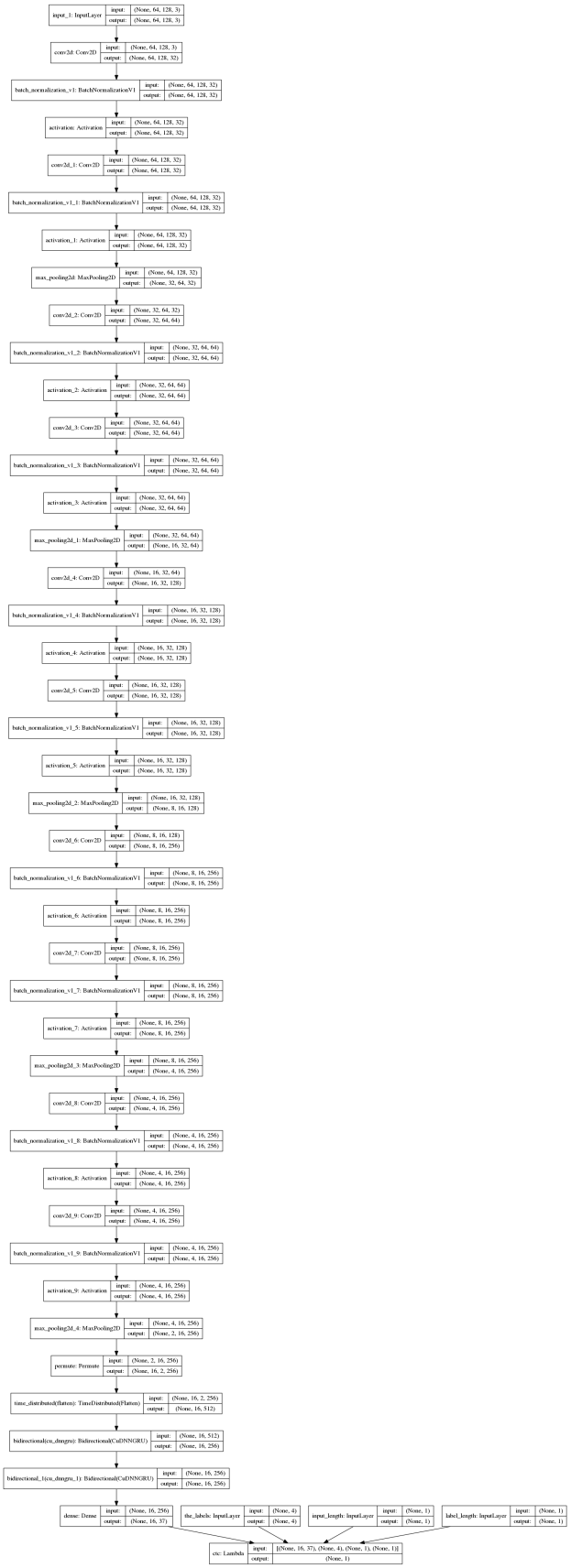

图 23. CTC 时序网络结构

D. 多位 mnist 数据集

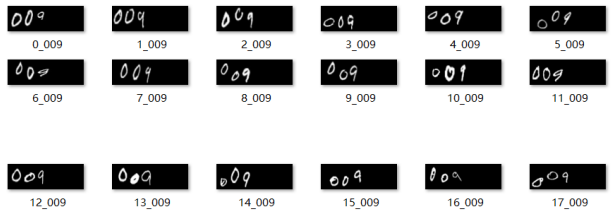

图 24. 多位 MNIST 图片生成

实现源代码参考https://github.com/shaohua0116/MultiDigitMNIST