Solosis

基于 LeNet-5 的手写数字识别

李明翰

数学与统计学院 2201 班, 华中科技大学

2022 年 11 月 27 日

目录

项目简介

简介

用 C++ 实现用于识别手写数字的 LeNet-5 卷积神经网络，采用 MNIST 作为训练集和测试集进行训练，并使用 OpenCV 进行图像支持。

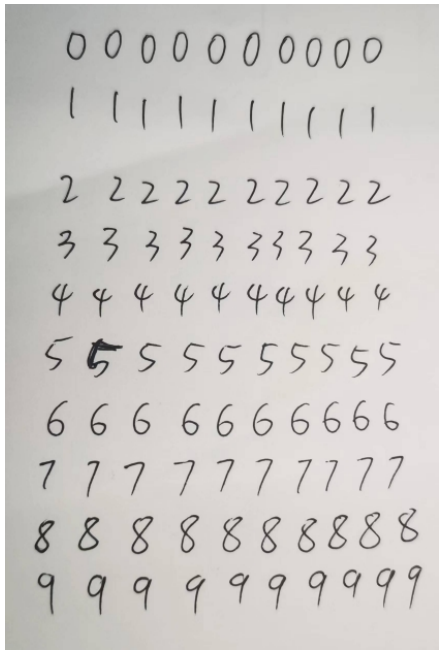
输入

一张图片，内容为一张写有黑色数字的白纸。

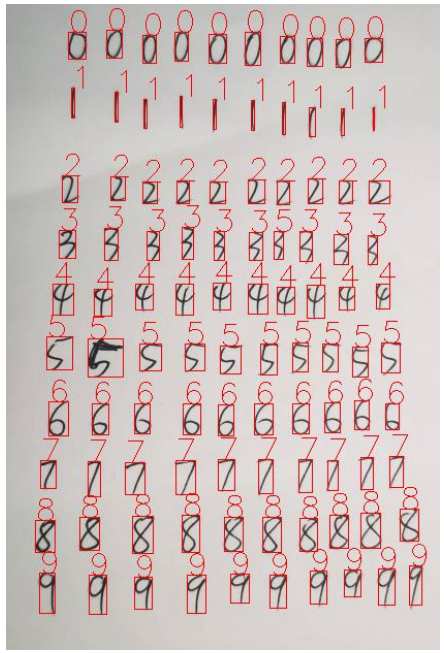
输出

识别后的图片。

效果



(a) 样例输入



(b) 样例输出

图: 一组样例输入输出

LeNet-5 模型概述

简介

LeNet-5 是一个卷积神经网络模型，可判断一个 32×32 大小的灰度图上所写的数字为 $0 \sim 9$ 中的哪一个。

输入

一个 32×32 的矩阵 $\mathbf{A}$，其中 $A_{i,j} \in [0, 1]$ 表示第 i 行第 j 列的像素点的亮度。

输出

一个整数 $y \in [0, 9]$，表示其预测的数字。

LeNet-5 模型概述

警告

由篇幅所限，模型结构、训练算法等均不展开阐述，不建议没有基础的同学学习这一部分。
本项目提供了已经训练过的模型和各种函数，可以直接使用。

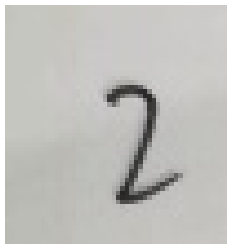
对单个数字进行分类

1. 灰度化
2. 亮度离差标准化至 $[0, 255]$
3. 阈值修改
4. 检测图形的边框并裁剪
5. 放缩为 32×32 大小
6. 再次亮度离差标准化
7. 将其作为输入提供给 LeNet-5 模型，得到输出

灰度化

即将彩色图片变为灰色图片。

如果需要，将图片反色，使得图片为黑底白字。



(a) 处理前

(b) 处理后

亮度离差标准化

记最小、最大亮度分别为 $\min, \max$，定义 $\mathrm{round}(x)$ 表示 x 四舍五入得到的整数。若 $\min \neq \max$，则对像素点 $A_{i,j}$，改变其亮度为

$$\mathrm{round}\left(\frac{A_{i,j} - \min}{\max - \min} \times 255\right)$$

也就是将 $[\min, \max]$ 线性映射到 $[0, 255]$，避免亮度不统一的情况。

(c) 处理前

(d) 处理后

阈值修改

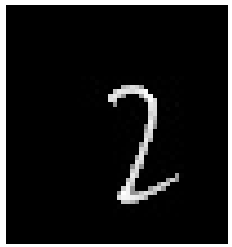
对每个像素点，若亮度小于 C（C 为常数），将其亮度设置为 0。

避免具有微弱亮度的像素点对模型判断的影响。

提示

本项目中取 $C = 130$。

(e) 处理前



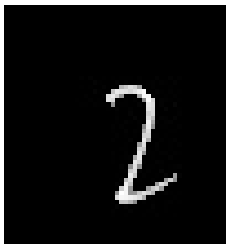
(f) 处理后

检测图形的边框并裁剪

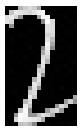
边框：最小的、能包含所有亮度非 0 的像素点的矩形。

检测边框，裁剪出子图。

避免书写位置的影响。



(g) 处理前



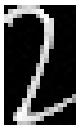
(h) 处理后

放缩为 32×32 大小

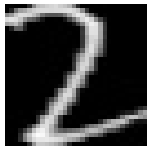
避免书写大小不同的影响。

提示

此处放缩为非比例放缩。
尽管强制放缩得到的图片可能很奇怪（如数字 1 放缩后可能是一张几乎全亮的图片），但只需要保证多张同一数字的图片，经过放缩后得到的图片相近即可。



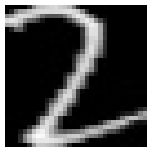
(i) 处理前



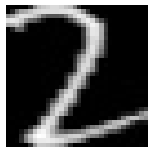
(j) 处理后

再次亮度离差标准化

避免放缩后导致亮度变暗的情况。



(k) 处理前



(l) 处理后

对整张图片进行分类

预处理

先灰度化，然后进行阈值修改。

判断边界

对每一个有亮度的点进行深度优先搜索，搜索出整个联通块，提供给 LeNet-5 作为单张图片进行判断。
其中点 (i, j) 和点 (x, y) 联通，当且仅当两个点的亮度非 0 且 $\max\{|i-x|, |j-y|\} \leq D$，其中 D 为常数。

提示

本项目中取 $D = 6$。

项目结构概述

项目根目录文件夹 `solosis` 下共有：

- 文件夹 `doc`：储存说明文档。
- 文件夹 `include`：储存 C++ `.h` 头文件。
- 文件夹 `mnist-data`：储存用于训练的 MNIST 数据。
- 文件夹 `model-data`：储存模型数据。
- 文件夹 `source`：储存 C++ `.cpp` 源代码。
- 文件 `.clang-format`：clang-format 格式化配置文件。
- 文件 `main.cpp`：C++ 主程序。
- 文件 `makefile`：指定编译方式的 Makefile。

LeNet-5 使用方法

`include/lenet.h` 提供了一个名为 `Lenet` 的类，拥有已经封装好的接口，可以直接调用。

初始化

`Lenet::Lenet(const char *filepath)`：从 `filepath` 读入模型参数。

使用

`int Lenet::predict(float in[32][32])`：将 `in` 作为输入，返回模型预估的数字。
其中 `in[i][j]` 应当为 $[0,1]$ 间的实数，背景色为 0，字迹色为 1。

LeNet-5 使用样例

```
#include "lenet.h"
#include <iostream>
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int main() {
    Lenet lenet = Lenet("model-data/standard.model");
    Mat image = imread("test-digit.jpg");
    float in[32][32];
    for (int i = 0; i < 32; i++)
        for (int j = 0; j < 32; j++)
            in[i][j] = image.at<uchar>(i, j) / 255.0;
    cout << lenet.predict(in) << endl;
    return 0;
}
```