Report of Handwritten Digit Recognition

手写数字识别南师附中组

2024 年 1 月 14 日

1 Group Members and Contributions

熊闻野：+4, 吴玉寒：-2, 王子枫：-2

Contributions:

熊闻野：I have completed the implementation and optimization of all the algorithms for the project and the control circuit for taking photos. I processed the MNIST dataset during the first two classes and generated the training set. In the following classes, I learned various functions in OpenCV online and applied them to perform operations such as denoising, dilation, and adding black borders to real images. These operations made the test images more similar to the training set for better segmentation. In the later sessions, I implemented the code for displaying digits on a 7-segment display in my_function. Finally, I built and wrote the code for the camera control circuit. Faced with issues such as noise, lack of clarity, and unnecessary parts in the captured images, I further learned additional functions of OpenCV online and achieved perfect processing of real photos.

王子枫：When we encounter difficulties, for instance, while coding the noise reduction part and the picture dividing part, I actively participated in information searches to address these difficulties. Additionally, I contributed to constructing the circuit responsible for displaying the recognized numbers. During the last lesson, I presented several ideas, including ways to ensure camera stability and organize the paper neatly to minimize script edges in the taken photo, which aimed to enhance the

quality of the photos we captured, and ultimately improving the accuracy of our recognition of numbers.

吴玉寒： During the project, I mainly contribute to the construction of the circuit of the seven-segment display and the bottom. Additionally, I not only came up with several methods to improve the image quality physically, but also assisted with the programming process, such as information searching and offering some ideas to improve the accuracy of the recognition. I also contributed to the optimization of the code.

2 Summary of the Project

This project consists of the following modules and here are their functionalities:

2.1 Code Section:

NumDect_CreateTrainSetOPENCV: Segment MNIST dataset into 5000 images of size 20x20, each containing a digit. Reshape these images into 1x400 arrays and add corresponding labels to create a training set.

TrainingData: Store processed MNIST dataset for future training with the KNN algorithm.

UserData: Store images to be tested.

my_function: Include functions for image cropping, LED display code, and photo-taking control code.

NumDect_ProcessRealImage: Main program with functions for environment initialization, importing the training set, training with KNN, taking photos, and processing images (grayscale, denoising, binarization, cropping, dilation, adding borders) to increase similarity between test and training sets, thereby improving recognition accuracy. Display recognition results on a 7-segment display.

2.2 Hardware Section:

Control Circuit: The main component is a button connected to a GPIO port and a GND port. Pressing the button triggers the takephoto function in my_function, initiating camera previewing, photo-taking and activating the LED light.

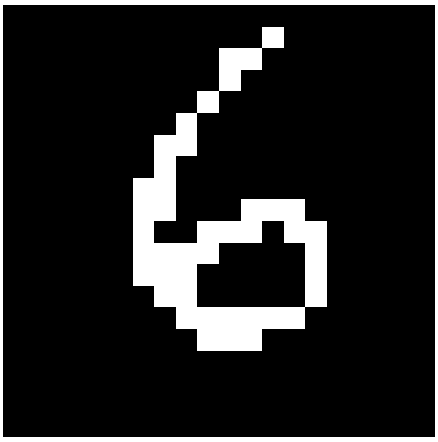
LED Circuit: The LED is connected to VCC and a GPIO port, with a resistor to protect the circuit. Pressing the button triggers the takephoto function in my_function, and the LED automatically lights up until the photo-taking is complete.

7-Segment Display Circuit: Utilize a 7-segment display with 7 segments connected to different GPIO ports and VCC. Individual control of these segments makes it displays digits of 0-9.

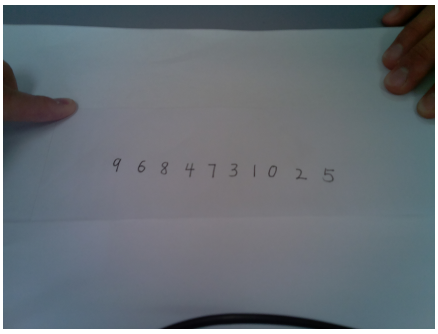
3 Results and Self-directed Exploration

Results:
The accuracy of our algorithm is around 90%, and the final result of the given numbers set is 80%

Here are some pictures of the results:

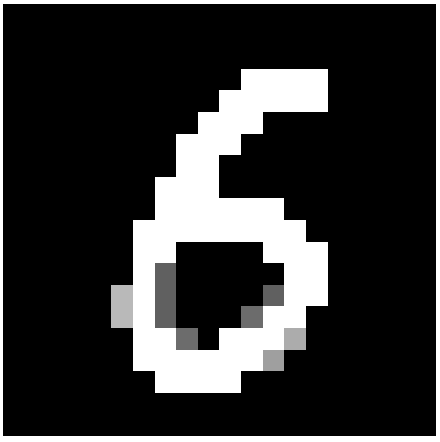


(a) Example of the training set(after reshape process)



(b) Photos of the numbers to be recognized

(c) Test numbers after image processing



(d) Example of the test set(after reshape process)

(e) Final result

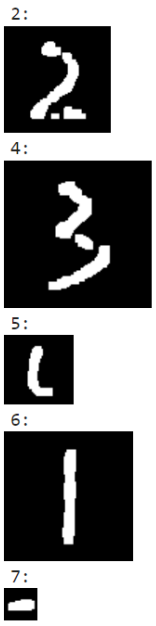
Result of Self-directed Exploration:

1. We used the adaptivethreshold instead of traditional threshold given in the Jupyter Notebook to do the binarization processing the numbers. This is because when taking photos, the light and shadow differs in different area. Traditional threshold method will make the processed image full of noise.

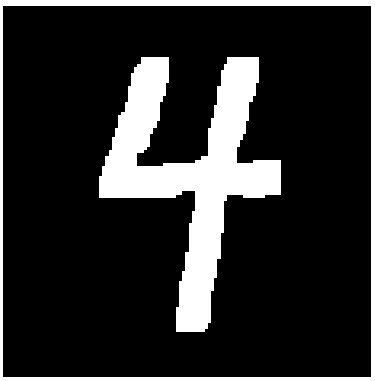
2. We used the dilate function to make the numbers more clear.

3. We used the morphologyEx function to denoise and underline the figure of numbers in the photo. Without the process, there will be much noise and breakpoints in the image of numbers

Here are some pictures to show the result of our self-directed exploration:



(f) Without self-directed processing



(g) With self-directed processing

Note that the 5th, 6th and 7th images without self-directed processing are actually parts of the digit 4, which should have been like the image beside.

4 Problems Encountered and Solutions

Problems Encountered:

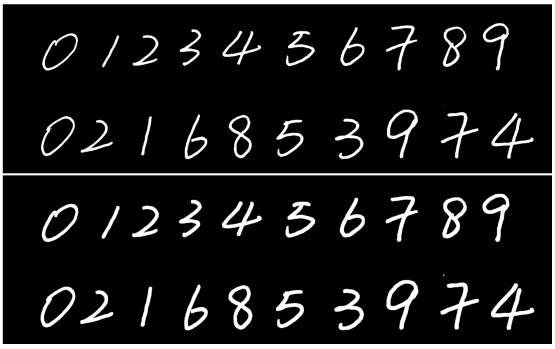
1. In the third and fourth classes, we found that the test set obtained was significantly different from the data used for training. This is partly due to the thinning of digital lines and image distortion caused by resizing the image. On the other hand, The split algorithm in my_function does not reserve a black edge for the image. These issues have resulted in very low recognition accuracy.

2. When we were doing the recognition of multiple lines, we often output excess parts due to the presence of noise in the image. Sometimes even causing program errors.

3. When we were doing the final task, we found that the photo taken by our camera was of deficient quality. There were many noise points and uneven brightness in the image. Also, the camera has a fixed focal length, so we had to take photos from a certain distance, which resulted in photos covering a lot of irrelevant interference(such as black power line and script edges)

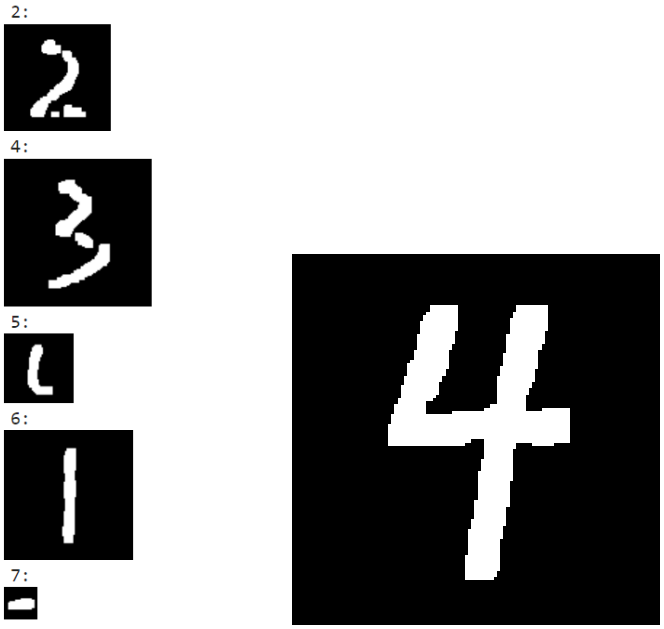
Solutions:

1. We used the OpenCV dilate function and numpy pad function to clarify the numbers and add black boarders. Here is the Comparison Chart



(h) Comparison Chart of the image with and without dilate function

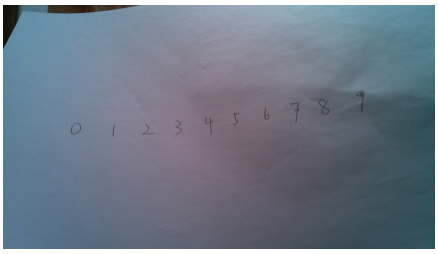
2. At first we tried to recognize noise points from digits by measuring their sizes. But we found it would make the program much slower and was unable solve the problems of large noise points and breakpoints in numbers. So after learning online, we used the morphologyEx function to denoise and underline the figure of numbers in the photo. The results indicated that its effect was perfect.



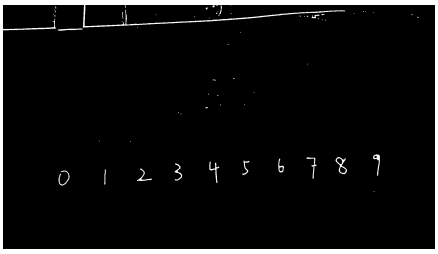
(i) Without self-directed processing (j) With self-directed processing

Note that the 5th, 6th and 7th images without self-directed processing are parts of the digit 4, which should have been like the image beside.

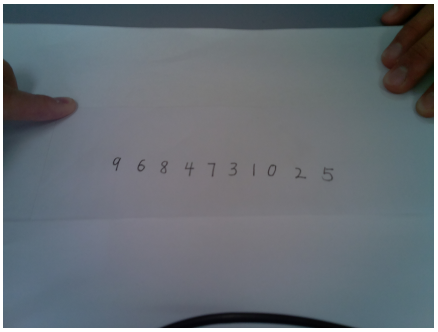
3. Facing the problem of uneven brightness, we used the adaptivethreshold instead of the given traditional threshold to do the binarization processing the numbers. Moreover, in order to remove distractions such as power lines that were captured, we accurately cropped the image. We also tried to improve the image quality physically, including build up a bracket to ensure camera stability and organize the paper neatly to minimize script edges in the taken photo.



(k) Photos before processing without physical improvement



(l) Photos after processing without physical improvement



(m) Photos before processing(adaptivethreshold and cutting) with physical improvement

(n) Photos after processing(adaptivethreshold and cutting) with physical improvement

5 Reflections and Future Improvements

1. Unfortunately, we have not fully resolved the issue of interfering substances being captured in images. An effective algorithm should be able to remove distractions such as power lines and creases, rather than relying on clipping. After diving deep into OpenCV library, we found that using Canny edge detector and OpenCV findCountours function would perfectly solve the problem.

2. The recognition rate is stuck at around 90%, which is close to the launch of traditional image processing algorithms and KNN algorithms. For higher recognition rates, we may consider using pre trained ResNet weights for processing

6 Personal Experience

Personal Experience:

熊闻野： Through this project, we have gained a preliminary understanding of artificial intelligence, image processing, and computer vision. Through in-depth learning of the OpenCV library and KNN algorithm, we have also enhanced our ability to proactively obtain information according to practical needs and make reasonable use of it.

吴玉寒： Through the project I gain the basic knowledge of how to use a Raspberry Pi and the way to control other devices using GPIO. Also, I learned the basic process of image capturing and recognition.

王子枫： The project underscored the significance of collaboration. For example, when we worked on the circuit building part, effective communication was essential to identify the GPIO pins corresponding to those on the Nixie tube. This collaboration bridged the gap between coding and circuit construction. Besides, to enhance team efficiency, it is necessary to allocate task at the beginning, or it may be a drawback to our workflow.