Development of OCR mobile application including miss-recognized proofreading system using database search algorithm

Ken Kariya^a, Takahiro Fujishima^a, Lifeng Zhang^{a,*}

^a Kyushu Institute of Technology, Kitakyushu, 804-8550, Japan

* Corresponding Author: zhang@ele.kyutech.jp

Abstract

In this paper, we introduce miss-recognition proofreading algorithm for optical character recognition(OCR) based on a database search method, implement the algorithm as an Android application, and verify its operation. We consider the proposed algorithm to have robustness against miss-recognition by searching the database partially using the character recognition results.

The proposed algorithm selects several characters randomly take out of OCR results, search the database which is prepared in advance for several times using selected characters, and decide the character string with the highest number of search as the proofreading results.

Also, we implement this algorithm as the Android OCR application. This application consists of five components as follows. (1) Image input (2) Pre-processing (cropping, denoising, and binarization) (3) Character recognition (using Tesseract-OCR software developed by Google and Hewlett Packard) (4) Correct miss-recognition (using the proposed algorithm), (5) Texts output

In this research, we present the recognition accuracy when we use the proposed algorithm and not use. Although it depends on the recognition accuracy of the OCR itself, our results reveal that the recognition rate is improved by the proposed method.

keywords: OCR, miss recognition, proofreading, database.

1. Introduction

In recent years, with the advanced internet technology, the logistic system has changed a lot, and international trade has broadly spread. As a result, the information printed on the paper needs to be converted to digital data at the receiver side.

Therefore, the use of character recognition in industrial applications is increasing. When converting the printed characters to character codes, manually input them to the computer or use a technology called Optical Character Recognition(OCR) are the usual two ways. In some case, OCR software does an excellent job, but some problems and challenges can occur as well. For example, usually, OCR with images produced by scanners gives high accuracy and excellent performance. However, acquire images with a scanner, it is necessary to pay the introducing cost for such a device. Besides, in the logistic process, it is usually difficult to read a delivery slip attached to a product or package with a non-portable scanner. For this reason, there is a high demand for a character recognition system implemented as a smartphone application. However, images produced by smartphone cameras usually are not as good of input as scanned images to be used for OCR due to the environmental or camera related factors as follows⁽¹⁾

- (1) Character settings (Fonts and Font sizes)
- (2) Print conditions (Blurring, Resolution)
- (3) Recognition conditions (Lighting, Rotation)

Currently, None of the exist OCR software that achieves 100% recognition. Therefore, it said that miss recognition is an unavoidable problem if using OCR software to finish a character recognition task. For this reason, reducing the miss recognition of OCR in such an environment is an active research area.

There are two approaches to reducing miss recognition. One is the prevention of miss recognition, and the other is the proofreading of miss recognition. The former is an approach of reducing miss recognition by re-learning the character which OCR could not recognize correctly. This approach provides some improvement in recognition accuracy, but people have to verify what kind of miss recognition will occur,

and needs the knowledge of OCR software as well. On the other hand, the latter is an approach of outputting the most probable character string from the characters and the arrangement of characters, even if recognition results include a miss recognition. Although this method requires the user to register a correct character string, they do not need to verify the miss recognition. This approach can improve the recognition accuracy if a good proofreading algorithm is implemented.

Therefore, in this research, we also proposed the verification algorithm using a database, implement OCR system as an Android application, and verify the recognition accuracy.

2. Principle and Structure

2.1 Application Structure

Figure 1 shows the process flow of the application we create in this research. There are five parts that struct this application.

Fig. 1. Flow chart of the application

(a) Image Input

First of all, we obtain an input image with the camera device. In this research, we use Pixel 3(Google) main camera and get images of 1980 pixel heights and 1080 pixel widths.

(b) Pre-processing

In this process, we convert the input image to be a suitable image. This process consists of three parts: Cropping, denoising, and binarization.

(c) OCR

we use OCR software, Tesseract. Tesseract⁽²⁾ is open source software for character recognition developed by

Google and Hewlett Packard. In this process, we obtain recognition results.

(d) Proofreading

With the algorithm shown in 2.3, recognition results are corrected. This algorithm uses a pre-created database for proofreading.

(e) Output

Finally, we obtain the output as a text format.

2.2 Pre-Processing

To eliminate unnecessary information or noise in an input image, we need to convert the image to the suitable format. Pre-processing reduces the redundant data and noise. We can enhance the effectiveness and easiness for an image to be processed in the OCR phase. In this research, we convert an input image by using cropping, non-local means denoising,⁽³⁾ and adaptive Gaussian thresholding.⁽⁴⁾

Cropping is one of the most basic image processing: Removing unwanted outer areas from an input image. After this process, we obtain an image that has only an area for recognition. Figure 2 shows that the screenshot of our application when we crop the image. It crops the area enclosed by the pink rectangle.

Fig. 2. Cropping process

To remove the noise contained in the input image, we use the technique called non-local means denoising. The method base on a simple principle: Replacing the color of a pixel with an average of the colors of similar pixels. However, similar pixels are not necessarily close to each other. Therefore, we should scan a vast portion of the image in search of all pixels that resemble the pixel one wants to denoise. Denoising is then done by computing the average color of these most similar pixels. The similarity is evaluated by comparing a whole window around each pixel, and not just the color.

After Denoising, we binarize the input image. When we create a binarized image from a grayscale image, threshold

processing is necessary. However, binarization with a single threshold can be suboptimal, particularly if the page background is of uneven darkness. Thus, we use an adaptive Gaussian threshold with the weighted average value of the close region of the target pixel as threshold. Gaussian function is used for weighting.

$$f(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \quad (1)$$

Fig. 3. Denoising and Binarization

2.3 Database Search Algorithm

To proofread the recognition results including miss recognition, we propose database search algorithm.

Fig. 4. Process flows of the proofreading algorithm

Figure 4 shows the process flow of proposed algorithm. The process flow consists of three parts. First, it randomly extracts several characters from recognition result. At this time, this algorithm stores not only the characters but also arrangement of characters as features of strings. Second, it searches the database for strings that contain the extracted characters in order. And it counts the number of times of search for each string. The propose algorithm is to repeat these steps several times and finally output the most frequently

searched string. Figure 5 shows a specific example of the correction process.

Fig. 5. An example of proofreading

Where "1INDICATOR" is an OCR result. First, it randomly extracts some characters from the OCR result ("N, A, R", "1, D, T"). Second, it searches database with extracted characters. In this case, strings containing characters in the order of "N, A, R" or "1, D, T" are searched.

3. Experimental Result and Discussion

In this research, we input an image containing the character strings of Figure 3 by a camera into our application, and recognized the character strings using the obtained image. Figure 6 shows the OCR results(Left) and proofreading results without using denoising and binarization(Right).

INDICATOR 800*900 INDICATOR 800*1100 INDICATOR 275 (25) INDICATOR 550*1400*25 INDICATOR 1100*1400*25 SIGNWEIGHT (HDG) SIGNWEIGHT (S GREEN) MA STAND-G MA STAND-Y F-18AEM F-18AD DESIGN FENCE PANEL (DS) -M CLAMP-FENCE BAND BEAUTY BASE CONE BASE ASH CAN W/CUP STEEL PATES-A	INDICATOR 800*900 INDICATOR 800*1100 INDICATOR 275(25) INDICATOR 550*1400*25 INDICATOR 1100*1400*25 SIGNWEIGHT(S GREEN) SIGNWEIGHT(S GREEN) MA STAND-G MA STAND-G F-18AEM F-18AD DESIGN FENCE PANEL(DS)-M CLAMP-FENCE BAND BEAUTY BASE CONE BASE ASH CAN W/CUP STEEL PATES-A
---	--

Fig. 6. OCR and proofreading results

Among the 17 character strings included in Figure 6, 4 strings were correctly recognized without any mistakes. After proofreading process, the number of correct strings increased 4 to 15.

Figure 7 shows the OCR results(Left) and proofreading results(Right) with using proposed denoising and binarization.

INDICATOR 800*900	INDICATOR 800*900
INDICATOR 800*1100	INDICATOR 800*1100
INDICATOR 275(25)	INDICATOR 275(25)
WDICATOR 550*1400*25	INDICATOR 550*1400*25
INDICATOR 1100*1408*25	INDICATOR 1100*1400*25
SIGINWBGHT(HDG)	SIGNWEIGHT(S GREEN)
SIGINWHIGHT(S GRBEN)	SIGNWEIGHT(S GREEN)
MA STAND-G	MA STAND-G
H/L STANDUY	STEEL PATES-A
F-18ABM	F-18AEM
F-18HD-	F-18AD
DBSIGN FBWCH PANBL(DS)-M	DESIGN FENCE PANEL(DS)-M
CLBMP-FBNCB BAND	CLAMP-FENCE BAND
BBAUTY BASB	BEAUTY BASE
CQNB BASB	CONE BASE
ASH-CAN W/CUP	ASH-CAN W/CUP
-WBLHFATES-A--	STEEL PATES-A

Fig. 7. OCR and proofreading results

Among the 17 character strings included in Figure 7, 5 strings were correctly recognized without any mistakes. After proofreading process, the number of correct strings increased 5 to 15.

In the results stated above, there was little difference between recognition accuracy with and without denoising and binarization. We think that this is because the denoising and binarization system included in the OCR software worked well. Besides, these results revealed that our proofreading algorithm can correct miss recognition.

4. Conclusions and Future Work

In this paper, we have presented miss-recognition verification algorithm for optical character recognition(OCR) based on database search method and implemented OCR system, implemented the algorithm as an Android application, and verified its operation. Although it depends on the recognition accuracy of the OCR itself, our results revealed that the recognition accuracy is improved by the proposed method.

In the next future we are going to improve the algorithm to be higher recognition accuracy.

References

- (1) Karez Abdulwahhab Hamad, Mehmet Kaya Hamad : "A Detailed Analysis of Optical Character Recognition Technology", International Journal of Applied Mathematics, Electronics and Computers, Vol. 4(Special issue), pp.244-249, 2016.09.03
- (2) Ray Smith and Google Inc. : "An overview of the Tesseract OCR Engine", ICDAR'07 Proceedings of the Ninth International Conference on Document Analysis and Recognition, Vol. 2, pp.629-633, 2007

- (3) Antoni Buades, Bartomeu Coll, and Jean-Michel Morel : "Non-Local Means Denoising", Image Processing On Line, Vol. 1, pp.208-212, 2011
- (4) G Deng, and L.W.Cahill : "An adaptive Gaussian filter for noise reduction and edge detection", Nuclear Science Symposium and Medical Imaging Conference, Vol. 3, pp.1615-1619, 1993