Improving accuracy of Tesseract in extraction of serial numbers from images of Counterfeit Electronics

Zarana Parekh[1,2], Chris A. Mattmann[1,2], and Karanjeet Singh[1,2]

[1]University of Southern California, Los Angeles, CA 90089 USA

Email: mattmann@usc.edu

[2]Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109 USA

Email: chris.a.mattmann@jpl.nasa.gov

1. INTRODUCTION

Tesseract is an open-source optical character recognition (OCR) engine with high character accuracy [2]. OCR engines are software packages that automatically recognize text in images, and extract that text. Tesseract supports text recognition in several languages such as English, French, German, Spanish, etc. Though Tesseract's support for text extraction has placed it well amongst the current state of the art in the area, its performance with extraction of alphanumeric text has been poor.

Alphanumeric text (e.g., `[0-9a-zA-Z]`) is present in images that identify e.g., serial numbers, part numbers, barcodes, etc., for e.g., weapons, or electronic parts. These types of images are increasingly prevalent on the web, and better support for extraction of these alphanumeric codes from them would aid a number of pertinent national efforts including the DARPA MEMEX initiative focused on domain specific search. We present work in this paper that aims to improve Tesseract's results in serial number extraction from images of counterfeit electronics as a sample use case from this domain. The significance of the problem is two-fold: (1) Accurate OCR systems can be used to provide automated text entry into computerized systems and digitize a large set of images; (2) The extracted serial numbers can be used to determine if the electronics item is counterfeit. Counterfeiting poses a risk to consumer electronics, to the intellectual property of large businesses, and overall to the the safety of software and hardware systems.

2. BACKGROUND AND RELATED WORK

Tesseract's OCR mechanism is detailed in [3] [4]. We summarize it here. Tesseract expects input as a binary image with optional polygonal text regions defined. Tesseract's OCR mechanism is as follows. Adaptive thresholding segregates the pixel values into text and background based on pixel intensity. The result is a binary image with black text on a near-white background. Connected Component Analysis (CCA) is then used to inspect possible nesting combinations of character outlines from the stored component values. After CCA, text lines are determined from the blobs. There are two sub-steps here: first specific regions of the image are analyzed for fixed and proportional text and then a `chop-then-associate` method breaks the text lines into words based on character spacing. Finally, a static and an adaptive classifier perform word recognition. It is a two-pass process that recognizes each word which is then used as training data by the classifier for further classification.

Work has been done in training Tesseract to support other languages such as [5] and [1] but these approaches do not involve image processing which can be tailored to identify text areas and character outlines more easily. Also, our approach involves an auto-rotation algorithm which uses radon transform to detect the text lines and determine the angle by which the image should be rotated to make the text horizontal.

3. APPROACH

Our serial number extraction process is detailed in this section. The first step in our approach is the pre processing of images. This involves the application of image processing tools to make images OCR-friendly without losing image quality. The image is modified as follows: (1) Setting optimum resolution to reduce noise in the image; (2) Binarize the image to improve the contrast between the text and background; (3) Image auto-rotation to deskew the image for better baseline identification; (4) Filter using linear interpolation; and finally (5) Re-sizing to magnify the text in the image. This approach is shown graphically in Fig. 1.

Tesseract has been trained with serial number fonts which includes samples of alphabets and digits as units for training data. Training comprises of feeding labeled images to Tesseract containing different serial number formats and generating a `language pack` which can be used later for OCR. An example of the training data for Tesseract is shown in Fig.2.

In the next section, we describe our evaluation and results.

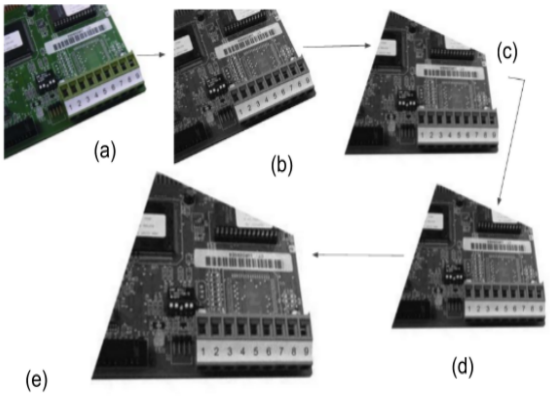

Figure 1: The original image has been processed for better readability: (a) original image (b) image resolution (c) auto-rotation (d) filtering (e) re-sizing

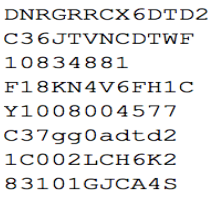

Figure 2: An example of training image containing serial numbers

4. TESTING AND RESULTS

The improved serial number trained version of Tesseract was tested with a mix of processed images from the domain of counterfeit electronics and from a seed list of web sites known to contain these images in different image formats, e.g., `png, jpg, etc`. The output was compared character-wise with the expected results and used to build the confusion matrix for different character classes as shown in Table 1. An example of a test image is shown in Fig. 3.

Table 1: Confusion matrix for Tesseract before and after modification

Actual character	Predicted character: Before modification											Predicted character: After modification										
	0	1	2	3	4	5	6	7	8	9	*	0	1	2	3	4	5	6	7	8	9	*
0	22	1									12	31										5
1		21	1								16	1		30								7
2			11								6	2			13							4
3				10							10	3				15						2
4					7						9	4					10					6
5						9					11	5						13				7
6						2	4				5	6						2	9		1	
7		1						5			6	7								9		2
8									6		6	8									7	4
9										5	6	9									7	3

* - unrecognised character

The results from Table 1 show that the characters are recognized with higher accuracy and occurrences of non-classified characters decreases. Character accuracy has been calculated by the following formula: $\frac{recognized\_characters}{total\_characters}$. Based on this formula, the overall results for character accuracy are as shown in Table 2. As can be seen from Table 2, character accuracy improves considerably for both classes.

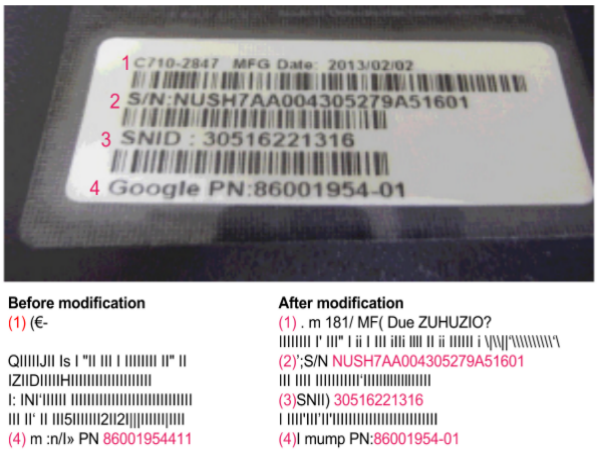

Figure 3: Comparison of results for a test image

Table 2: Character accuracy of Tesseract before and after modification

	Class: a-z	Class: 0-9	Overall
Before modification	0.52	0.56	0.52
After modification	0.61	0.77	0.73

5. CONTRIBUTION

The contributions of our work are three-fold. First we are improving serial number extraction from counterfeit electronics images which will be integrated into the national scale DARPA MEMEX initiative. Second, the pre-processing and trained data we have leveraged has been integrated into the open source Apache Tika toolkit to increase its adoption and use. Finally, the language pack that we trained and created for serial number recognition has been integrated into the Homebrew deployment of Tesseract so others can immediately benefit from our work.

Acknowledgments

This work was supported by the DARPA XDATA/Memex program. In addition, the NSF Polar Cyberinfrastructure award numbers PLR-1348450 and PLR-144562 funded a portion of the work. Effort supported in part by JPL, managed by the California Institute of Technology on behalf of NASA.

6. REFERENCES

[1] S. Rakshit and S. Basu. Recognition of handwritten roman script using tesseract open source ocr engine. *arXiv preprint arXiv:1003.5891*, 2010.

[2] S. V. Rice, F. R. Jenkins, and T. A. Nartker. The fourth annual test of ocr accuracy. Technical report, Technical Report 95, 1995.

[3] R. Smith. An overview of the tesseract ocr engine. 2007.

[4] R. Smith. Tesseract ocr engine. *Lecture. Google Code. Google Inc*, 2007.

[5] N. White. Training tesseract for ancient greek ocr, 2013.