Text recognition approaches for indoor robotics: a comparison

Obadiah Lam, Feras Dayoub, Ruth Schulz, Peter Corke

ARC Centre of Excellence for Robotic Vision, Queensland University of Technology * †

Abstract

This paper evaluates the performance of different text recognition techniques for a mobile robot in an indoor (university campus) environment. We compared four different methods: our own approach using existing text detection methods (Minimally Stable Extremal Regions detector and Stroke Width Transform) combined with a convolutional neural network, two modes of the open source program Tesseract, and the experimental mobile app Google Goggles. The results show that a convolutional neural network combined with the Stroke Width Transform gives the best performance in correctly matched text on images with single characters whereas Google Goggles gives the best performance on images with multiple words. The dataset used for this work is released as well.

1 Introduction

We live in a sea of information. We are surrounded by text which we read almost unconsciously while we are moving around our environment. It blends into our lives to the point where we stop appreciating how valuable it is. Granting the same reading abilities to a mobile robot would improve the level of its performance and interaction with humans to a great extent. These words and symbols are cues the robot can use for localisation and navigation. In addition to that, the text information can be used by the robot to create web queries which provide relevant information to augment its current sen-

Fig. 1: Overall Process: the input image is cropped and fed into one of four text recognition algorithms. The results are shown in the bottom row of boxes.

sory data. It is an attractive source of information which should not be ignored.

Reading text from scanned documents, optical character recognition (OCR), has come a long way and its accuracy has reached a point where it is able to recreate even the format of the original document including fonts, colours and the layout. However, recognising text in unconstrained images is still an open problem. The problem arises from multiple factors including variation of light conditions, cluttered background, and the height, font, size, orientation, colour and texture of the text.

In this paper we compare multiple approaches for text extraction from natural images (see Fig. 1) and we show that the use of a convolutional neural network for optical character recognition provides promising results when combined with the state-of-the-art text detection methods. We investigate whether the current performance of current unconstrained text recognition approaches is sufficient in an environment where the text is floor-to-ceiling and has challenging texture and lighting conditions.

The rest of the paper is organised as follows. In Section 2, we discuss related work in the field. Section 3 gives an overview for the different text recognition methods used in this work. Section 4 presents the experimental set-up and the dataset used for evaluation. Finally we draw conclusions and discuss future work in Section 6.

2 Related Work

There is a large body of literature with many approaches proposed to extract text information from natural images. Generally these approaches focus on one stage in the text recognition process, with only a small proportion of systems addressing the full text recognition pipeline. The first stage is detecting text within the image and finding its location. This text is then passed into the character recognition stage, where the individual letters and numbers in the detected text are recognised. Finally, a word detection stage takes the sequence of recognised characters and corrects it, often with a dictionary of possible words.

Recently, a fast method for text detection was introduced in [Epshtein et al., 2010] called the stroke width transform (SWT). The input image is first converted to grayscale and then an edge detector is used to produce a binary edge map. Parallel lines are then detected and used to calculate stroke width for each pixel. Pixels with similar stroke width are grouped together to form a single character. The method is sensitive to the quality of the extracted edges which in-turn depend on the level of noise and blur in the input image.

Edge-based features have also been used successfully for text detection in [Chen et al., 2004]. The method is aimed at video frames and it utilised an added step to verify detected text regions using machine learning (ML) approaches. Two ML methods were tested, multilayer perceptrons (MLP) and support vector machines (SVM), to verify candidate text regions. In [Yao et al., 2007], a method based on geometrical features from connected components is used to train an SVM in order to be classified into characters or non-characters. In [Neumann and Matas, 2012], colour and geometrical features from detected maximally stable extremal regions (MSERs) [Matas et al., 2004] were used to classify the candidate text regions into character and non-character.

More details about other approaches for text detection methods can be found in [Zhang and Kasturi, 2008].

After detecting the text regions in an image, the next step is character recognition. Two approaches emerge in the literature to handle this step. The first one takes the output of the detection step and binarises it then feeds it to a traditional Optical Character Recognition (OCR) engine such as Tesseract [Smith, 2007], which has a long history in the application of OCR and has been used very successfully for a very large scale document understanding on a massive scale [Vincent, 2007]. However, when it comes to natural images, the results of our earlier work in [Posner et al., 2010] showed its accuracy is low and severely compromised by the variation of possible textures. The success of this earlier work relied on a probabilistic error correction algorithm to correct errors such as single character substitution, additional spaces, and missing spaces. Even with this correction algorithm 'texture words' were still found in images with no text, for example, from the textures in a brick wall.

The second approach treats the problem as an object recognition problem. One of the promising algorithms under this latter approach is the PhotoOCR algorithm [Bissacco et al., 2013] released recently by Google and quickly becoming one of the leading benchmarks. PhotoOCR uses Convolutional Neural Networks (CNN) [LeCun et al., 1998] trained with over 2 million images. Convolutional neural networks are also used in [Wang et al., 2012; Opitz et al., 2014] for the text detection stage. In [Ciresan et al., 2011] a committee of several neural networks vote on the output.

An alternative approach is to skip the character recognition stage entirely and use convolutional neural networks to perform word detection. This has been shown to be successful for large dictionaries of tens of thousands of words [Jaderberg et al., 2014].

We would like to point out that most existing work focuses on improving the performance of text extraction for specific datasets such as those from ICDAR (International Conference on Document Analysis and Recognition) 2003, 2005 and 2011. This has driven an improvement in performance by a few percentage points on these datasets but the general problem is still unsolved. In this work we are not aiming to solve the general problem of text information extraction from natural images but rather achieving reliable performance in the context of our own environment (i.e. door labels and directional signs on a university campus, see Fig. 4).

3 Approach

In this section we describe the text recognition techniques used in this paper. First, we describe our method, which combines standard methods for detecting regions of text within an image with a Convolutional Neural Net-

work for character recognition. We also give an overview of the benchmark techniques used: two different modes of Tesseract (an open-source OCR package) and Google Goggles (an image-based search app for Android devices).

3.1 Stroke width Transform and convolutional neural network, SWT-CNN

Our method uses the Minimally Stable Extremal Regions detector (MSER) for text detection, then the Stroke Width Transform (SWT) for character segmentation, and a Convolutional Neural Network (CNN) for character recognition. Word detection is not implemented.

The method begins by using some of the 'characterness' cues outlined in [Li *et al.*, 2013] to detect and subsequently extract the text from the image. First, the Minimally Stable Extremal Regions detector is used as a region detector [Matas *et al.*, 2004]. The Stroke Width Transform is performed on the regions that were detected with MSER [Epshtein *et al.*, 2010], skeletonised to improve computation speed. Regions with a consistent stroke width that satisfy weak geometric constraints such as aspect ratio are tagged as text.

The recognition phase is performed with a CNN which has been trained on the 74k dataset [de Campos *et al.*, 2009]. This dataset is labelled with 62 classes of upper case letters, lower case letters, and digits. The synthesised computer font subset of the dataset was used, comprising 62992 character images. Due to the similarity between characters such as 'l' and '1', aggregate classes have been used, resulting in a total of 46 classes (see Table 1 for a list of the aggregated classes). The aggregate classes were chosen to deal with the scaling problem of character cases. That is, without prior knowledge of expected scale or context from surrounding characters, lower and upper case characters such as 'c' and 'C' are visually identical.

We use a convolutional neural network architecture proposed by [Simard *et al.*, 2003]. The CNN has 4 layers, with 2 convolutional layers and 2 fully-connected layers. The sizes of the layers was 5-50-100-46. The output layer is the set of 46 classes, with the chosen class for a given input being the output neuron with the highest value.

The CNN was trained for 25 iterations (epochs) on the character images in the 74k font dataset with small random affine distortions. Training used backpropagation with an initial learning rate of 0.001, decreasing by a factor of 0.794 every 4 epochs. At this point, the CNN was then trained for 3 additional epochs on non-distorted training images at a constant learning rate of 0.0002. The performance on the training set was 3.3% error.

The output of the recognition phase is the individual character that was recognised in each of the text regions detected in the image. In future implementations of the method, a word recognition stage will choose the correct character out of the aggregated options based on the context of the surrounding characters.

3.2 Tesseract

We use two modes of the open-source Tesseract OCR Engine [1] [Smith, 2007] as benchmarks for text recognition. Tesseract, like most available OCR packages, has been developed for integration with word-processing tools rather than robots. Text found in images obtained from robots often has geometric distortion. Tesseract can deal well with such skewed baselines and has been used successfully for OCR in images obtained by robots, when combined with an additional probabilistic error correction algorithm [Posner *et al.*, 2010].

As Tesseract was designed for performing OCR on scanned printed text, poor text recognition performance is the result unless text detection has been initially performed on the images. We manually crop the images around the text visible in the image to simulate a text detection stage and pass the cropped text regions into Tesseract. We use two of the available page segmentation modes: Mode 4 (the default mode which assumes a single column of text of variable sizes) and Mode 6 (which assumes a single uniform block of text, similar to the test images in the cropped dataset images).

3.3 Google Goggles

Google Goggles [2] is an image-based search app for Android devices running Android 2.2 and above that enables users to search the web using photos. Google Goggles attempts to find useful information about the submitted image, returning web results, similar images, and text. In this experiment, we only consider the text results returned. Preliminary experimentation with Google Goggles indicated that it is very successful with text detection in natural images. As Google Goggles does not currently have an available API for use, an Android device is required. Each image in the dataset must be manually loaded into the Android application and the results must be manually recorded.

4 Experiments

In this section we describe the dataset and experimental setup.

---

[1] "tesseractOcr", http://code.google.com/p/tesseract-ocr/, last accessed on 07-August-2014

[2] "Google goggles", http://www.google.com/mobile/goggles/, last accessed on 07-August-2014

Tab. 1: List of aggregate classes. Each column is aggregated to the label on the top row.

0	1	C	J	K	M	N	P	S	U	V	W	X	Y	Z
O/o	L	c	j	k	m	n	p	s	u	v	w	x	y	z

Fig. 2: Example images where all the methods fail

Fig. 3: Example images where only our method has succeeded

Tab. 2: The output for an image containing a longer piece of text (29 characters)

Approach	Output	Edit Distance	Success
Ground Truth	QUT Z BLOCK BUSINESS BUILDING	0	1.0
Our approach	a T r HU0GK BUSI ESS BUILhDV G	11	0.62
Tesseract (Mode 4)	BUSINESS BUILDING	12	0.59
Tesseract (Mode 6)	z MK 3H?L"b"uEuiSicSs	26	0.10
Google Goggles	QUT Z BLOCK BUSINESS BUILDING	0	1.0

4.1 Dataset

A series of 81 photos were taken around QUT Gardens Point Campus with text in each image. The images contained either building names (single letters e.g. 'S'), floor levels (digits e.g. '4'), room names (letter followed by room number e.g. 's408'), or longer pieces of text including words (e.g 'QUT Z BLOCK BUSINESS BUILDING' as in Figure 9). As Tesseract expects the images to be cropped around the text, cropped images were used for all techniques to allow a fair comparison (see Fig. 4 for an example image and its three cropped images). We manually crop the images instead of using an automatic bounding box method such as a sliding window approach to ensure the regions are correctly cropped and contain the label text.

The resulting dataset of cropped images has 103 images containing text[3]. The cropped images were classified, according to the type of text visible in them, as Class 1: one or two characters (building names and floor numbers), Class 2: door labels (a single letter followed by three or four numbers), and Class 3: strings of text containing multiple words.

4.2 Experimental setup

The 103 cropped images were presented to each of the four approaches, with the resulting text outputs recorded. The text result, t, was compared with the ground truth, g, using Levenshtein edit distance [Levenshtein, 1966], $D(g,t)$. We define a measure of success, $S(t,g)$ as:

$$S(t,g) = \begin{cases} 0, & \text{if } D(g,t) >= L(g) \\ (L(g) - D(g,t))/L(g), & \text{if } D(g,t) < L(g) \end{cases}$$

[3] The dataset is available at http://tinyurl.com/ot6ypab

Fig. 4: An example image and its three cropped images in the dataset (Ground Truth: 'Computer Lab', 'Z412', 'Z412')

where $L(g)$ is the length of the ground truth. This success measure scales with $L(g)$ to allow comparison of results between images with ground truths of different string lengths.

From this performance measure, we classified each result as 'failed', 'partial', or 'match':

$$\begin{aligned} \text{Failed} &\quad S(t,g) = 0 \\ \text{Partial} &\quad 0 < S(t,g) < 1 \\ \text{Match} &\quad S(t,g) = 1 \end{aligned}$$
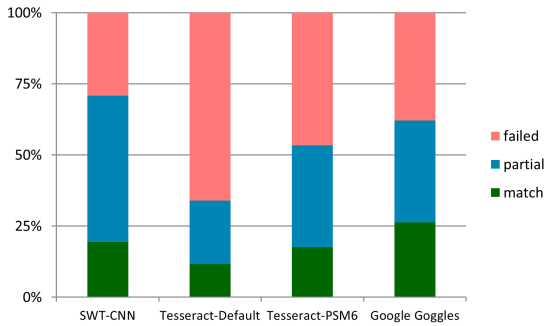

Fig. 5: Overall performance

5 Results

For the 103 cropped images, SWT-CNN matched 20, Tesseract (Mode 4) matched 12, Tesseract (Mode 6) matched 18, and Google Goggles matched 27 (see Fig. 5). Our approach had an additional 53 images that were partial matches, with only 30 failed, while the failures for
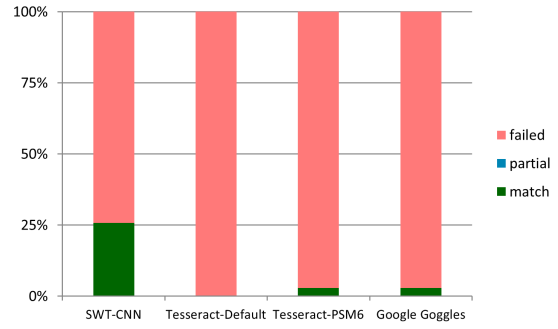

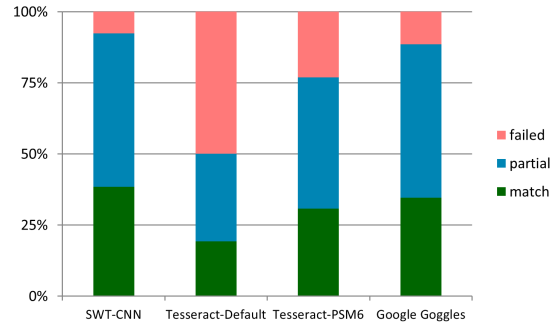Fig. 6: Images containing one or two characters
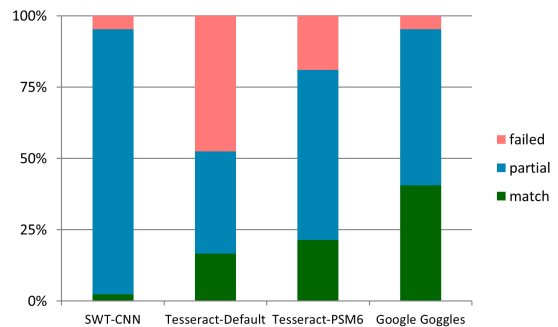

Fig. 7: Images of door labels

Fig. 8: Images containing strings of text with multiple words

Fig. 9: Example of an image with a long string (multiple words) of text

the other methods were 68 for Tesseract (Mode 4), 48 for Tesseract (Mode 6), and 39 for Google Goggles.

Our method outperformed the other approaches on Class 1 images (one or two characters), with 9 matches out of 35 images, while the other approaches had at most one match (see Fig. 6 and 3). Class 1 images for which all methods failed often had issues such as large geometric distortion or extra lines within the character (see Fig. 2).

Performance on Class 2 images (door labels) was similar between our approach and Google Goggles which both outperformed Tesseract: our approach and Google Goggles had fewer 'failed' results than the Tesseract approaches (see Fig. 7). All approaches had many partial matches for the 26 images, with 14 partials for our approach, 8 for Tesseract (Mode 4), 12 for Tesseract (Mode 6), and 14 for Google Goggles. Common errors in the door label images for all approaches include replacing the letters for numbers and vice versa, for example replacing 'S' with '3' or '5'. Given an appropriate lexicon of expected formats for room names, these errors could be reduced.

Google Goggles outperformed the other approaches on Class 3 images (strings of text), with 17 matches out of 42 images, 23 partials, and only 2 failures (see Fig. 8). Our approach had only one match, but had 39 partial matches and only 2 failures. Once again, the majority of the results returned were partial matches because of the character ambiguity mentioned previously, as well as the lack of a lexicon correction stage.

A typical example of an image containing longer strings of text was Figure 9, where the ground truth 'QUT Z BLOCK BUSINESS BUILDING' contained 29 characters. Google Goggles achieved a match for this image, and our approach, Tesseract (Mode 4), and Tesseract (Mode 6) achieved a partial match, with edit distances of 11, 12, and 26 respectively (see Table 2). Our approach found many of the individual characters correctly, and potentially could be corrected if the word recognition step was included. Tesseract (Mode 4) found two of the five words, probably due to differences in font and size between the other words. Tesseract (Mode 6) found only a few of the characters correctly.

Google Goggles is a text recognition tool with high performance rates, but as there is no API or description of how it works (although [Bissacco *et al.*, 2013] might be relevant), it is not available for use in a robotic platform. However, the results provide an existence proof that these performance levels can be reached.

These results show that our method performs well compared to benchmarks, particularly for difficult single character text recognition.

6 Conclusion and Future work

We have evaluated the performance of different text recognition techniques for the application of label text recognition by a mobile robot in an indoor environment. The tested methods were: our own approach using standard text detection methods (Minimally Stable Extremal Regions detector and Stroke Width Transform) combined with a convolutional neural network, two modes of Tesseract, and the experimental mobile app Google Goggles.

Our early work, using a convolutional neural network combined with the Stroke Width Transform, correctly recognises the most text in the single characters category. Google Goggles gives the best performance on long strings of text containing multiple words. Neither of these methods can currently achieve practical performance on our dataset, which includes scenes with complex lighting and translucent text. None of the approaches are currently practically useful for robotic applications but their performance can be improved by restricting font and text size in the urban environment.

We are currently working on implementing our method for our mobile robot. This is part of a larger project to detect and extract semantic information from its surroundings and utilise this information to improve navigation in unknown environments.

Future work will also look at different convolutional neural network architectures and larger training sets for better character recognition. A word recognition stage will be added to our text recognition method.

References

[Bissacco *et al.*, 2013] Alessandro Bissacco, Mark Cummins, Yuval Netzer, and Hartmut Neven. PhotoOCR: Reading text in uncontrolled conditions. In *IEEE International Conference on Computer Vision (ICCV), 2013*, pages 785–792. IEEE, 2013.

[Chen *et al.*, 2004] Datong Chen, Jean-Marc Odobez, and Herve Bourlard. Text detection and recognition in images and video frames. *Pattern Recognition*, 37(3):595–608, 2004.

[Ciresan *et al.*, 2011] Dan Claudiu Ciresan, Ueli Meier, Luca Maria Gambardella, and Jürgen Schmidhuber. Convolutional neural network committees for handwritten character classification. In *International Conference on Document Analysis and Recognition (ICDAR), 2011*, pages 1135–1139. IEEE, 2011.

[de Campos *et al.*, 2009] Teo de Campos, Bodla Rakesh Babu, and Manik Varma. Character recognition in natural images. *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*, 2009.

[Epshtein *et al.*, 2010] Boris Epshtein, Eyal Ofek, and Yonatan Wexler. Detecting text in natural scenes with stroke width transform. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010*, pages 2963–2970. IEEE, 2010.

[Jaderberg *et al.*, 2014] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv:1406.2227*, 2014.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[Levenshtein, 1966] Vladimir Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics-Doklady*, 10:707–710, 1966.

[Li *et al.*, 2013] Yao Li, Wenjing Jia, Chunhua Shen, and Anton van den Hengel. Characterness: An indicator of text in the wild. *arXiv:1309.6691*, 2013.

[Matas *et al.*, 2004] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767, 2004.

[Neumann and Matas, 2012] Lukas Neumann and Jiri Matas. Real-time scene text localization and recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012*, pages 3538–3545. IEEE, 2012.

[Opitz *et al.*, 2014] Michael Opitz, Markus Diem, Stefan Fiel, Florian Kleber, and Robert Sablatnig. End-to-end text recognition using local ternary patterns, mser and deep convolutional nets. In *11th IAPR International Workshop on Document Analysis Systems (DAS), 2014*, pages 186–190. IEEE, 2014.

[Posner *et al.*, 2010] Ingmar Posner, Peter Corke, and Paul Newman. Using text-spotting to query the world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2010*, pages 3181–3186. IEEE, 2010.

[Simard *et al.*, 2003] P.Y. Simard, D. Steinkraus, and John C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *7th International Conference on Document Analysis and Recognition (ICDAR), 2003. Proceedings.*, pages 958–963, Aug 2003.

[Smith, 2007] Ray Smith. An overview of the Tesseract OCR engine. In *9th International Conference on Document Analysis and Recognition (ICDAR), 2007*, volume 7, pages 629–633, 2007.

[Vincent, 2007] Luc Vincent. Google book search: Document understanding on a massive scale. In *9th International Conference on Document Analysis and Recognition (ICDAR), 2007*, volume 2, pages 819–823. IEEE Computer Society, 2007.

[Wang *et al.*, 2012] Tao Wang, David J Wu, Adam Coates, and Andrew Y Ng. End-to-end text recognition with convolutional neural networks. In *21st International Conference on Pattern Recognition (ICPR), 2012*, pages 3304–3308. IEEE, 2012.

[Yao *et al.*, 2007] Jin-Liang Yao, Yan-Qing Wang, Lu-Bin Weng, and Yi-Ping Yang. Locating text based on connected component and svm. In *International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR'07), 2007*, volume 3, pages 1418–1423. IEEE, 2007.

[Zhang and Kasturi, 2008] Jing Zhang and Rangachar Kasturi. Extraction of text objects in video documents: Recent progress. In *International Workshop on Document Analysis Systems (DAS), 2008*, pages 5–17, 2008.