Mobile Scanner and OCR (A first step towards receipt to spreadsheet)

Clement Ntwari Nshuti, *cntwarin@stanford.edu*

***Abstract*—In the recent years we have seen an explosion of advances in technologic systems for finance management and payment. Electronic payment systems with credit cards and other systems such as *Venmo* and *Apple Pay* are part of our everyday lives. Moreover companies like *Mint* make electronic budgets and expenses management seamless. However, despite all these advances the paper receipt is still ubiquitous and there is no easy mean to convert it into an electronic format. In this paper we investigate a first step towards the digitalisation of paper receipts; namely we develop a pipeline for performing OCR on the picture of a document taken by a mobile phone.**

I. Introduction

SMARTPHONE penetration is growing at an astonishing rate and so is the quality of the cameras embedded in these devices. We investigate how to harness the power of these cameras to convert a smartphone into a scanner. Combined with an open source OCR system such as Tesseract [1], this would bring a powerful Scanner-OCR system at the tip of the fingers of users. Such a system could be used for several purposes, each with its challenges. The one presented in this paper is in the end goal of converting paper receipts to spreadsheets. Our work focuses on building a robust OCR system for pictures taken from a mobile phone. As we hypothesize that given a high quality text format of a given receipt, one can add an NLP system to segment the products and prices components. We present a system that is robust against background color, cluttered background and documents that are folded or crinkled. In section II we describe the approach taken along with the challenges against with our system is robust and its limitations. Section III is dedicated to our evaluation procedure and section IV to the results of this evaluation. Finally in section V we present our opinion on the next steps to take.

II. Approach

A. Hypotheses

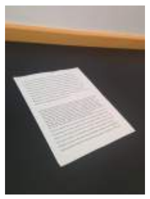
We make the following assumptions about the pictures taken by the user.

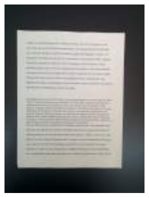
1) The document to be scanned is a white rectangular single paper. Our goal being to process receipts, most receipts if not all fall into this category
2) This document is the largest rectangular shape on the picture.
3) The picture is not blurred. This assumption is based on the idea that the user should be able to take the best picture possible of the receipt.

B. Challenges

The assumptions above leave the following situations to which we want our system to be robust against.

1) **Camera angle**. We assume that there are three possible angles from which the photos can be taken: side,top, front. They are illustrated on figure 1
2) **Background color and clutter**. Detecting edges of the document might be harder depending on the brightness and patterns in the background as illustrated on figure 2.
3) **Document quality** Finally a last challenge is if the document is folded or crinkled as illustrated on figure 3.


(a) Side


(b) Top

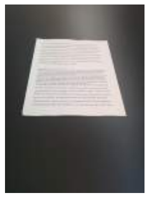

(c) Front

Fig. 1: One challenge for a mobile scanner is that pictures of the document can be taken from different angles. In this work we make the assumption that pictures can be taken from three angles (side, top, front). And assess the perfomances for each of these possible angles.

C. Algorithm

Before being fed to the Tesseract OCR system [1] the user's pictures is first preprocessed to produce a binary image with a scanned effect. First the image is denoised using a gaussian blur. The window size of the denoising filter is 5×5 by default but we recommend using a 9×9 filter when the background is noisy. The image is then, optioannally sharpened. This

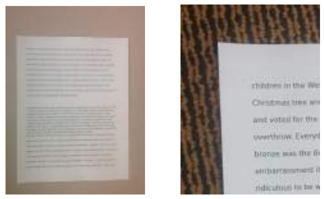


Fig. 2: Another challenge is the background color and clutter. A backgroung to bright might make it harder to detect edges of the document. The same holds for a background that is too cluttered.

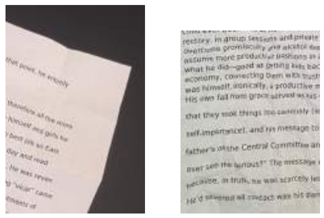


Fig. 3: The last element against which we want to be robust is the quality of the document. Whether it is folded or crinkled, we want our system to perform well.

sharpenning step is only necessary when the color of the background is too close to the document's color. By default this step is not necessary.

After sharpening we use a canny edge detector with $[75, 200]$ as the thresholds. This parameter can be kept as it is. It did not influence very much the performance of the OCR.

From the result of the Canny edge detector, we extract the largest quadrilateral. This largest quadrilateral is obtained by approximating all closed regions in the output of the edge detector by quadrilaterals and keeping the largest from these.

This quadrilateral is the extracted from the original (undenoised and unsharpened) image and warped into a straight rectangle. This result is then binarized using adaptive threshold.

The binarized image is fed to Tesseract. This algorithm is largely inspired from Adrian Rosebrock's algorithm [2] and is summarized in figure 4

III. Evaluation procedure

The quality of the OCR depends on the quality of the adaptive thresholding step. Parameters in this step are the window size (`w_adapt_bin`) and minimum threshold (`adapt_bin_thrs`). In the adaptive thresholding algorithm, if the average brightness in a tile is below `adapt_bin_thrs` the tile will be considered to be uniformly white. During the evaluation procedure, we looked for the values of these parameters that maximize the robustness of the scanner against the challenges descripted in II.

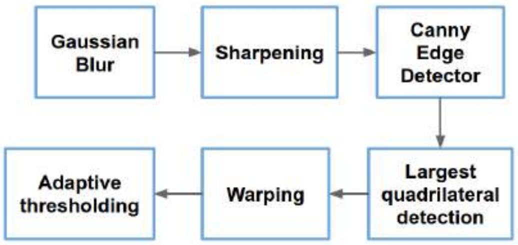


Fig. 4: To improve the quality of the OCR output, the input image is first preprocessed to yield a scanner effect.

A. Dataset

For each of the conditions below, we took 11 pictures of 11 different pages of a short story from Jonathan Franzen [3].

- **DARK**. The best conditions underwhich a picture can be taken is from the top with a dark background. This will be used as the baseline. Other configurations will be just a variation from this. We'll either vary the background color, the camera orientation or the document quality (see figure 1).
- **BRIGHT**. Pictures of the document from the top with a bright background (see figure 2).
- **NOISY** Pictures of the document with a background that has several small patterns (see figure 2).
- **SIDE** Pictures of the document from the side (see figure 1).
- **FRONT** Pictures of the document from the front (see figure 1).
- **FOLDED** Pictures of the document after being folded (see figure 3).
- **CRINKLED** Pictures of the crinkled document (see figure 3).

This gives a total of 7 conditions and 77 pictures.

B. Metrics

For each of the 7 conditions above, the 11 eleven pictures are fed through the scanning pipeline described in section II. The result is converted to a `*.txt` file by Tesseract. We then measure the cosine distance between the output document from Tesseract and the original page from the short story. Specifically the cosine distance is measured between the word distributions of each of the two documents. We chose this metric because, the OCR system is unlikely to change the order of words, the most common error will be not correctly identifying words. Therefore a metric such as the cosine distance which is independent of the word orders is suitable. For each of the scenarios above we computed the average cosine distance (called accuracy in the results section) as a function of the two design parameters `w_adapt_bin` and `adapt_bin_thrs`.

IV. Results

The size of the window used in adaptive thresholding doesn't have any significant influence on the OCR accuracy

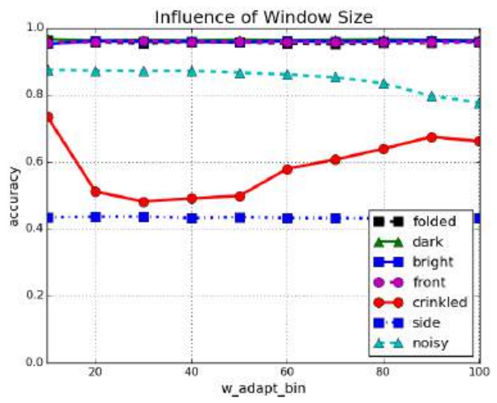


Fig. 5: Influence of the window size. Experiments performed with at threshold of 0.1.

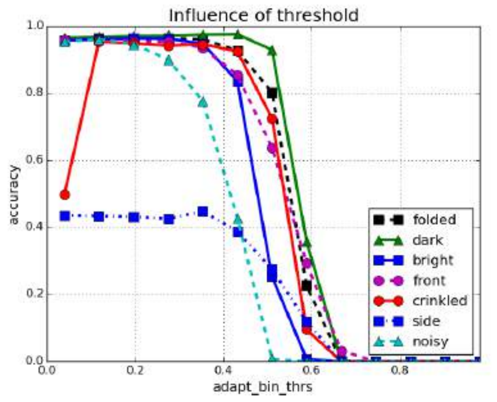


Fig. 6: Influence of the threshold. Experiments performed with a window size of 50×50

except for the crinkled documents as we can see on figure 5. The experiments on this figure were carried out with a threshold value of 0.1 times the maximum brightness 255.

The brightness threshold is significantly more influential than the window size. For values above 0.4 times the maximum brightness 255, the quality drops drastically. We can see that an optimal choice is around 0.3. This value gives indeed the best quality accross all scenarios. However we see that no value of the window size or of the threshold can give a good quality when the image is taken from the side. The text in the warped image is too distorted for the OCR to perform well.

V. Conclusion

We proposed a method for converting a picture of a document into a binary image which fed to the Tesseract OCR system achieves an accuracy of 98%. This system is robust against background color and clutter. And gives the best performances when the picture is either taken from the top of the document or from the front. The current implementation of the code is in MATLAB and requires to upload the image on a computer. One step of the improvements would be to implement the code on a mobile phone. Further improvements would also consist in implementing the NLP section that is dedicated to the process of receipts.

Acknowledgment

The author would like to thank the instructors for the most organized class they have ever taken at Stanford.

References

[1] R. Smith and G. Inc, “An overview of the tesseract ocr engine,” in *Proc. 9th IEEE Intl. Conf. on Document Analysis and Recognition (ICDAR*, 2007, pp. 629–633.

[2] A. Rosebrock, “How to build a Kick-Ass Mobile Document Scanner,” http://www.pyimagesearch.com/2014/09/01/build-kick-ass-mobile-document-scanner-just-5-minutes/, 2014, [Online; accessed 02-June-2015].

[3] J. Franzen, “The republic of bad taste,” http://www.newyorker.com/magazine/2015/06/08/the-republic-of-bad-taste, 2015, [Online; accessed 02-June-2015].