mOCRa: Mobile OCR Application*

mOCRa: Aplicación OCR Móvil

Xose R. De La Puente
IRLab, A Coruña Univ.
Campus Elviña s/n
A Coruña
xose.puente.romay@udc.es

Ismael Hasan
IRLab, ICT Centre
Campus Elviña s/n
A Coruña
ihasan@udc.es

Resumen: En los últimos años, los teléfonos móviles han evolucionado hasta convertirse en dispositivos con cámaras de gran resolución y conexión a Internet. En este contexto, surge la idea de aplicar tecnologías OCR a las fotos de los móviles. Esta idea origina *mOCRa*, la aplicación presentada en este trabajo.
Palabras clave: Reconocimiento óptico de caracteres, aplicación móvil, Android

Abstract: In the last years, mobile phones have evolved to devices with high-resolution cameras and Internet connection. In this context, the idea of applying OCR techniques to the pictures taken with these devices rises. As result of this idea, we have built *mOCRa*, the application presented in this work.
Keywords: Optical character recognition, mobile application, Android

1. Introduction

The evolution of mobile devices has been vertiginous: two decades ago they were big devices, with a low autonomy of battery, and they only could be used to make phone calls; nowadays, mobile phones are devices with multimedia capabilities, Internet access, etc. Their features include the integration of the mobile phones with digital cameras. With this in mind, it rises the idea of using a mobile device to extract the text from the pictures taken with the camera; this idea is materialised in *mOCRa*, and its first version for Android devices.

2. System Overview

mOCRa client application offers to the users an accessible and easy-to-use interface, optimised capturing of images with text and tools to manage and edit the texts recovered from the pictures.

The application interface includes an adaptable grid so the user can use it to align it with the lines of text (see Figure 1); it also allows to set the quality of the picture to be taken according to the amount of grid lines. In order to ease the use of the application, four quality levels have been defined in *mOCRa*: low, medium, high and very high. The "very high" level is suitable to analyse full page texts.

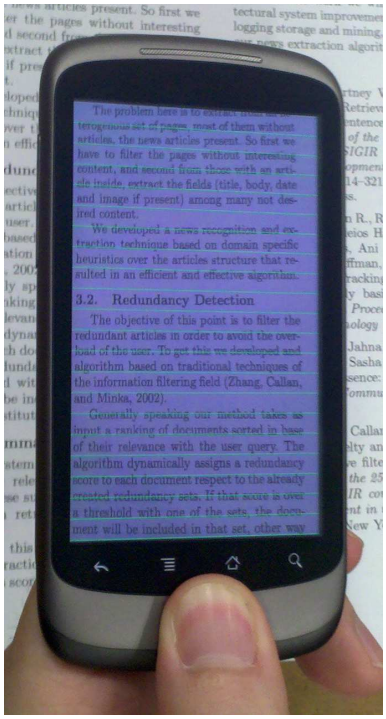

Figure 1: mOCRa working with a Nexus One device

The process of obtaining the text from a picture goes as follows: once an image is captured with the phone it is sent to the server. The server processes it and returns the text to

* This work was funded by FEDER, *Ministerio de Ciencia e Innovación* and *Xunta de Galicia* under projects TIN2008-06566-C04-04 and 07SIN005206PR.

the phone; the user of the device can modify the text using a text editor, can store it, can send it via e-mail or can select a portion of text to be used as a query against Google. A video demonstration of *mOCRa* can be found in the IRLab web[1].

3. System Architecture

The application *mOCRa* was developed for the Android platform. Also, it was designed to be easily migrated to other mobile operating systems. To accomplish this, the application follows a client-server architecture; moreover, to guarantee compatibility, communication issues are managed using Web Services. Both client and server systems follow a component-based architecture, to build a functionally scalable application.

The mobile device system comprises the following modules: user interface module, Web Services based communication management module, stored texts management module and public key management module, to communicate with the server using secure connections.

The server system comprises the following modules: image pre-processing module, OCR analysis module, a parsing module for each Web Service and a business logic module for each Web Service. The aforementioned Web Services are used to configure the application, to send the images and responses and to avoid sending data in the case the server is overloaded.

4. Evaluation

The application was evaluated in terms of effectiveness, to check that the results are correct, and efficiency, to check that the results are obtained in an acceptable time. The reference to be compared with was another mobile OCR application, *SnapIt*, available for Android systems and developed by *mocrsoft*. This system has been commercialised for a while, and it is one of the main competitors of *mOCRa*.

4.1. Effectiveness

In this analysis the similarity between the text obtained using the applications and the original text was measured using the Levenshtein distance (LD) (Levenshtein, 1966), in a similar way it was used in other evaluations of OCR systems (E. Borovikov and Turner, 2004), and the normalised Levenshtein distance (NLD).

Levenshtein distance is used to give an insight of the amount of differences between two sequences of characters. It measures the minimum number of operations needed to transform one of the sequences to the other. These operations include replacement, addition and deletion of single characters.

Normalised Levenshtein distance is used to give an insight about the similarity: a value of zero means that the sequences are completely different, and a value of one means that they are equal. Its formula is

$$NLD = 1 - \frac{LD}{LDMax} \qquad (1)$$

where $LDMax$ is the length of the longest text.

The testbed includes an heterogeneous set of texts to cover several significant characteristics to compare the applications. *Snapit* does not allow to load images from the memory of the phone, it only retrieves the text from pictures taken with the camera, and it does not store them. For this reason, the images used to compare the results are not exactly the same for both systems. To minimise the impact of using different pictures, several images were taken for each text and application, and in the evaluation it was used the one offering the best results for each text.

The applications were tested using two different Android devices, with different specifications: HTC Magic and Nexus One. To do the tests, three texts were used; a brief explanation of each text follows, accompanied with the comparison of the results for each application and mobile phone.

The first text used in the evaluation is an economics text written in English. It is a full page containing 3.729 characters, with a font size of 12pt. *Snapit* only allows to take pictures in landscape format, so it was neccessary to take two photographs to process the entire page. The "very high" quality option was used in *mOCRa*.

Table 1 shows that the results of *mOCRa* clearly outperform the ones obtained with *SnapIt*. To deal with full page images, *mOCRa* includes a "very high" image quality mode to optimise the results. *SnapIt* does not offer a similar option; moreover, due to the fact that the pictures can only be taken

[1] http://www.irlab.org/?q=publications/multimedia

in landscape format, it is necessary to do two photographs.

Metric	SnapIt		mOCRa	
	Nexus	HTC	Nexus	HTC
LD	2430	1830	54	923
NLD	0.34835	0.50925	0.98552	0.75248

Table 1: Full page text similarity

The second text contains 697 characters, uses a font size of 12pt, and it is written in English. This document contains the same phrase repeated with different font types and formats (boldface, italic and underlined). The phrase does not make sense, but it is intended to cover most of the usual characters: "The (quick) brown {fox} jumps! over the $3,456.78 <lazy> #90 dog & duck/goose, as 12.5 % of E-mail ".

The results in table 2 show that *mOCRa* again outperforms *SnapIt*. The configuration used in *mOCRa* was the "high" quality one, which is suitable to extract the text from several paragraphs.

Metric	SnapIt		mOCRa	
	Nexus	HTC	Nexus	HTC
LD	419	212	50	113
NLD	0.39885	0.69584	0.93084	0.84327

Table 2: Several fonts text similarity

A complex table was choosen as the document for the last comparison. Each cell contains several lines of text; the total amount of characters is 1387. The pictures include the two upper rows of the table and the full width (five columns), covering half of the page.

From the results showed in table 3 it can be inferred that nor *mOCRa* neither *SnapIt* apply layout analysis techniques to deal with tables or multi-column layouts. The results are pretty bad: the distribution of text causes problems in the OCR process and the lines of the tables are extracted as extra characters. In this test *mOCRa* results include more errors than the *SnapIt* ones (*mOCRa* has a higher LD); however, its NLD value is better. The main implication of this fact is that *mOCRa* extracts more text from the noise of the image than *SnapIt*, but it is more accurate obtaining the real text.

The tests also show an interesting fact: *SnapIt* offers better results with the HTC Magic device, despite the fact that its camera quality is lower that the Nexus One camera. Because of this, it is our belief that this application uses some image processing techniques

Metric	SnapIt		mOCRa	
	Nexus	HTC	Nexus	HTC
LD	1077	1072	1258	1309
NLD	0.22350	0.22711	0.26818	0.24640

Table 3: Table text similarity

which are dependant on the resolution of the pictures. This does not happen with *mOCRa*, offering better results as the quality of the images improves.

4.2. Efficiency

These tests were run in a Nexus One device. It accessed the web through a wireless 802.11g connection (54 Mb/s) to communicate with the server[2]. The execution times showed in tables 4 and 5 comprise the process since a picture is sent for processing until the text is shown in the mobile device.

4.2.1. *mOCRa*

The application allows the users to choose the quality (size) of the picture to send. This choice can be done by adapting the size of the display grid, or by selecting one of the four available predefined quality levels. In the tests the data was collected for these predefined levels. The computing time in *mOCRa* includes:

1. mobile - configuration sending,
2. mobile - image splitting,
3. mobile - creation and sending of Web Service packages containing the image,
4. server - image reconstruction,
5. server - image pre-processing,
6. server - OCR processing,
7. server - text sending,
8. mobile - text display.

Also, it is worthy to mention that the communications involving images and text are encrypted using SSL.

To obtain the results 10 pictures were processed. Table 4 shows the time results: it can be observed that the use of the "very high" option is very time-consuming, but it takes advantage of the maximum quality the camera can offer. As previously stated, *mOCRa* results improve with quality images.

[2]Intel(R) Core(TM)2 Quad CPU Q6600 @ 2.40GHz, 4 GB of ram

About the time results, we must remark that the process which most penalises the excution time of *mOCRa* is to send the image. This operation takes longer than the image pre-processing and retrieval of text.

mOCRa processing time (s)			
Image quality	Mean	Best time	Worst time
Low	6	5	8
Medium	8	7	9
High	9	8	15
Very high	16,9	14	25

Table 4: *mOCRa* processing time.

4.2.2. *SnapIt*

The tests run to check the efficiency of this application were the same used to check the efficiency of *mOCRa*. We cannot provide any information about the way in which *SnapIt* retrieves the text (we have no access to its source code), but we can assume that it also uses a server (since it cannot work without a connection to the Internet), so the process should share some similarities.

Table 5 shows the results. *SnapIt* times improve the times of *mOCRa*; therefore, it is our belief that this application does not take advantage of the quality of the cameras integrated in the mobile phones.

SnapIt processing time (s)		
Mean Time	Best Time	Worst Time
4,25	2,5	7

Table 5: *SnapIt* processing time.

5. Conclussions and Future Work

The application presented in this work, *mOCRa*, shows good results. The system can provide an excellent startpoint to build specific and complex systems, offering for instance generation of summaries, generation of snippets, entities detection or language translation.

Our next works with *mOCRa* will include the improvement of the results by the use of *top-down* layout detection techniques, table boundaries detection techniques and the use of text post-processing techniques to detect the noise and to correct bad-recognised words. With these improvements, the text could be used for complex Information Retrieval tasks: the techniques of Parapar, Freire, and Barreiro (2009) can be applied to use these texts in IR systems; moreover, the work of K. Taghva and Condit (1994) states that if a picture is good enough and a post-processing of the text is applied, the final result has the same quality as a text manually created and corrected.

Finally, it is worthy to mention that we are working on image compression techniques and in the improvement of the communication protocols to obtain better results in terms of efficiency.

Bibliografía

E. Borovikov, I. Zavorin and M. Turner. 2004. A filter based post-OCR accuracy boost system. In *Proceedings of the 1st ACM workshop on Hardcopy document processing*, pages 23–28, Washington, DC, (USA).

K. Taghva, J. Borsack and A. Condit. 1994. Results of applying probabilistic IR to OCR text. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 202–211, Dublin, Ireland.

Levenshtein, V.I. 1966. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Phys. Dokl.*, 10(8):707–710.

Parapar, Javier, Ana Freire, and Álvaro Barreiro. 2009. Revisiting n-gram based models for retrieval in degraded large collections. In *ECIR '09: Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, pages 680–684, Berlin, Heidelberg. Springer-Verlag.