OCR4JkanjiCards: Exploring Japanese Character Recognition

Oswaldo Bassani Neto, Celso Setsuo Kurashima, Marcelo Zanchetta do Nascimento
Universidade Federal do ABC
{oswaldo.bassani, celso.kurashima, marcelo.zanchetta}@ufabc.edu.br

Abstract

The OCR4JkanjiCards system is presented in this work, as a proposal for taking photos of Japanese characters using a smartphone and have them automatically recognized within a digital Japanese dictionary. Image processing techniques and neural network strategies are necessary, as well as the exploration of several programming tools.

1. Introduction

The increase in processing power and memory available on mobile devices like cell phones, smart-phones or handheld, allow user to have pocket dictionaries providing fast access to information.

The goal of this work is to explore a set of tools that allows joining several features in a unique system. Our OCR4JkanjiCards system is based on character recognition, and on the processing of photos taken from street plates, building facades and other texts in the real world.

2. System Overview

The OCR4JkanjiCards system basically works with an image of a kanji as the input and gives back the kanji identification as the output, through a user interface as shown in Figure 1. Kanji's are the complex characters used in Japanese language.

Figure 1: OCR4JkanjiCards User Interface.

The inner window in the upper left part of Figure 1 is a picture, taken with a cell phone camera, of a poster in the street containing a kanji character. After several image processing steps the kanji was clearly separated in the upper right inner window of Figure 1. In the lower part of this figure, the result of recognition process is shown in the line marked by the green box, in the form of ID codes of kanji's. These codes refer to a Japanese dictionary where we can obtain the meanings in English. In this case, we used the "Java Kanji Flashcards 500" [1] data base[1].

We created OCR4JkanjiCards system based on the project called OCR4J [2]. This is an open source library that allows the creation of applications with character recognition like OCR (Optical Character Recognition), but is limited to A-Z characters.

So our system started from the integration of the OCR system with the Kanji education software, and went to the development of image processing techniques for Kanji characters. We are interested in investigating how kanji's can be efficiently processed for recognition tasks, given that the character writing is complex and composed of non-connected lines.

Correlated work can be found in references [3] and [4]. The first one is a project of kanji dictionary for mobile devices.

3. Methodology

Since the purpose of this experiment is the use of images acquired from photographic cameras, we apply some image processing techniques in order to convert input images into binary image format before using the OCR system.

So, the initial step is to convert the input colored image to gray scale and to apply a median filter. Then, it is converted to a binary image, with an appropriate threshold. Finally, this image is passed through an

[1] "Java Kanji Flashcards 500", available in: http://nuthatch.com/java/kanjicards/, accessed in March 2008.

invert function to get it prepared for the kanji recognition process.

However, we have further removed excess of spurious granularity of the image that appeared in our experiments. This was done by applying a region growth to the image and manually removing unwanted regions.

The character recognition sub-system makes use of a neural network of multilayer perceptron type. This neural network can vary its architecture, for example we can change the number of input elements and the number of internal layers as well as the number of neurons on the layers. The only predetermined value is the number of output neurons, which must equal the number of characters we want to recognize.

Before training the neural network, however, we have realized that some kanji's are not fully identified by the neural network because they are composed by two or more sets of connected pixels. In the example shown in Figure 1, the left part of that kanji is disconnected from the right part of it.

For these cases, we propose to break the kanji into two sub-parts. By doing so, these two sub-parts can be easily recognized individually.

4. Results

Our experiments are still in the beginning and were carried out on commodities desktop PC's. We have trained a neural network with 17 kanji's. For each kanji we used two images of training for each one. We also added one kanji with two parts for training the neural network, i.e., with four more training pictures.

In the OCR4JkanjiCards, we have used an image-processing tool called ImageJ [5][6], which was easy to use in the software environment during the process that convert images into binary format.

In Figure 2 we show the example of the result of image processing techniques applied on the kanji with two parts. The leftmost image is the original input image. The other images to the right are the results of the steps of gray scaling, filtering, threshold, binary, invert, and spurious removing.

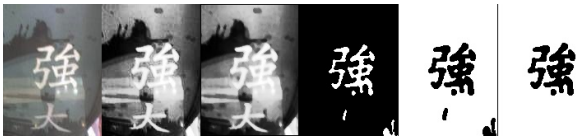

Figure 2: image processing steps of the OCR4JkanjiCards system.

All training images and test images passed through this image processing steps. And, after training the neural network we have got the result shown in Figure 1, where the kanji of two parts was successfully recognized. The same positive result was observed in all testing image for kanji recognition.

5. Conclusions

In this paper we propose the OCR4JkanjiCards system, which is a Japanese character (kanji) recognition tool based on the OCR4J environment. The goal is to use it on mobile devices like cell phones.

We have presented initial experiments with the OCR4JkanjiCards system. This includes our proposal for breaking kanji's into two parts during recognition process on the neural network, as a proof of concept. Consequently, not only the system can run faster but also the recognition becomes easier.

As future work, there are many more experiments to be carried. For example, to expand the quantity of kanji's to be trained on the neural network, and to measure the processing time on mobile devices.

In the image processing steps, we plan to investigate the detection of the base lines to find the characters, as observed by Koga *et al* [3].

6. References

[1] S. Wood Ryner, N. Chikamatsu, H. Nozaki, S. Yokoyama, S. Fukada - "Java Kanji Flashcard 500: Kanji, Java, and the World Wide Web", 16th annual Unicode Conference, Tokyo, 1998.

[2] Project: "Ocr4j", Optical Character Recognition library for Java, available in: http://sourceforge.net/projects/ocr4j/, accessed in March 2008.

[3] M. Koga, R. Mine, T. Kameyama, T. Takahashi, M. Yamazaki, T. Yamaguchi - "Camera-based Kanji OCR for Mobile-phones: Practical Issues", proceedings of the 2005 Eight International Conference on Document Analysis and Recognition (ICDAR'05), 2005.

[4] H. A. Rowley, M. Goyal, J. Bennett - "The Effect of Large Training Set Sizes on Online Japanese Kanji and English Cursive Recognizers", proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02), 2002.

[5] W. Bailer - "Writing ImageJ Plugins - A Tutorial", software documentation.

[6] Software: "ImageJ", available in: http://rsb.info.nih.gov/ij/, accessed in March 2008.