AUTOMATED CHEQUE PROCESSING OF INDIAN BANK CHEQUES

[1]Uma Karambelkar, [2]Vaishnavee V. Kulkarni,[3]Ankita V. Gandhi
Savitribai Phule,Pune University
Email: [1]uma.karambelkar@gmail.com, [2]kulkarnivaishnavee@gmail.com,
[3]ankitavg4@rediffmail.com

Abstract—Objective of this paper is to generate a system to process the standard Indian bank cheques in an automated way. Currently for cheque processing there is no automated system in India, as the automating the processing of handwritten cheque is a challenging task. We propose a solution for the same. The system will process the cheque image, perform OCR using open source Tesseract OCR engine, and validate the cheque. Depending upon the user's confirmation about the OCR text, the system will send the image for further processing. In contrast to the traditional cheque processing techniques, ACP makes this process easier, reduces the human efforts and provides convenience, better deposit availability.

Keywords: Automated Cheque Processing, android, handwriting recognition, OCR, Tesseract,

I. INTRODUCTION

Cheques represent a significant segment of payment instruments in India. Automated Cheque Processing system will provide a new way of processing bank cheques. For cheque processing and clearing following methods are used in India:

- *Cheque Truncation*: Truncation is a system to avoid the movement of the physical cheque issued by a drawer to the drawee branch. The physical document will be truncated and sent to the drawee branch as an electronic image along with the relevant information like their fields, date of presentation, presenting banks etc.

- *Electronic Fund Transfer*: EFT is a Scheme introduced by Reserve Bank of India (RBI) to help banks offering their customers money transfer service from account to account of any bank branch to any other bank branch in places where EFT services are offered.

- *Electronic Clearance System (Debit):* Electronic Clearing Service (Debit) scheme provides faster method of effecting periodic and repetitive payments by 'direct debit' to customers' accounts (duly authorized) thereby minimizing paper transactions and increasing customer satisfaction. Electronic Clearing Service (Debit) envisages "a large number of debits and one credit" in the case of collection of electricity bills, loan installments, Club fees etc. by the Utility Service Providers.

- Electronic Clearance System (Credit): ECS payments can be initiated by any institution (called ECS user) who have to make bulk or repetitive payments to a number of beneficiaries. For example mutual fund paying dividends to its unit holders, companies

- distributing dividends to their shareholders etc.

The ACP system will make the cheque clearing process easier as well as faster. The system will fetch the details of cheque by confining the Region of Interest (ROI) according to the Indian standard[2] for bank cheque. Fig. 1 shows the standard Indian cheque format.

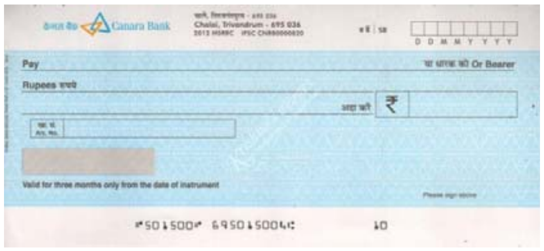


Fig. 1. Indian standard cheque format

Optical Character Recognition (OCR) will be performed on this Region of Interest and data will be read from cheque. According to the results of the OCR further transactions will be executed.

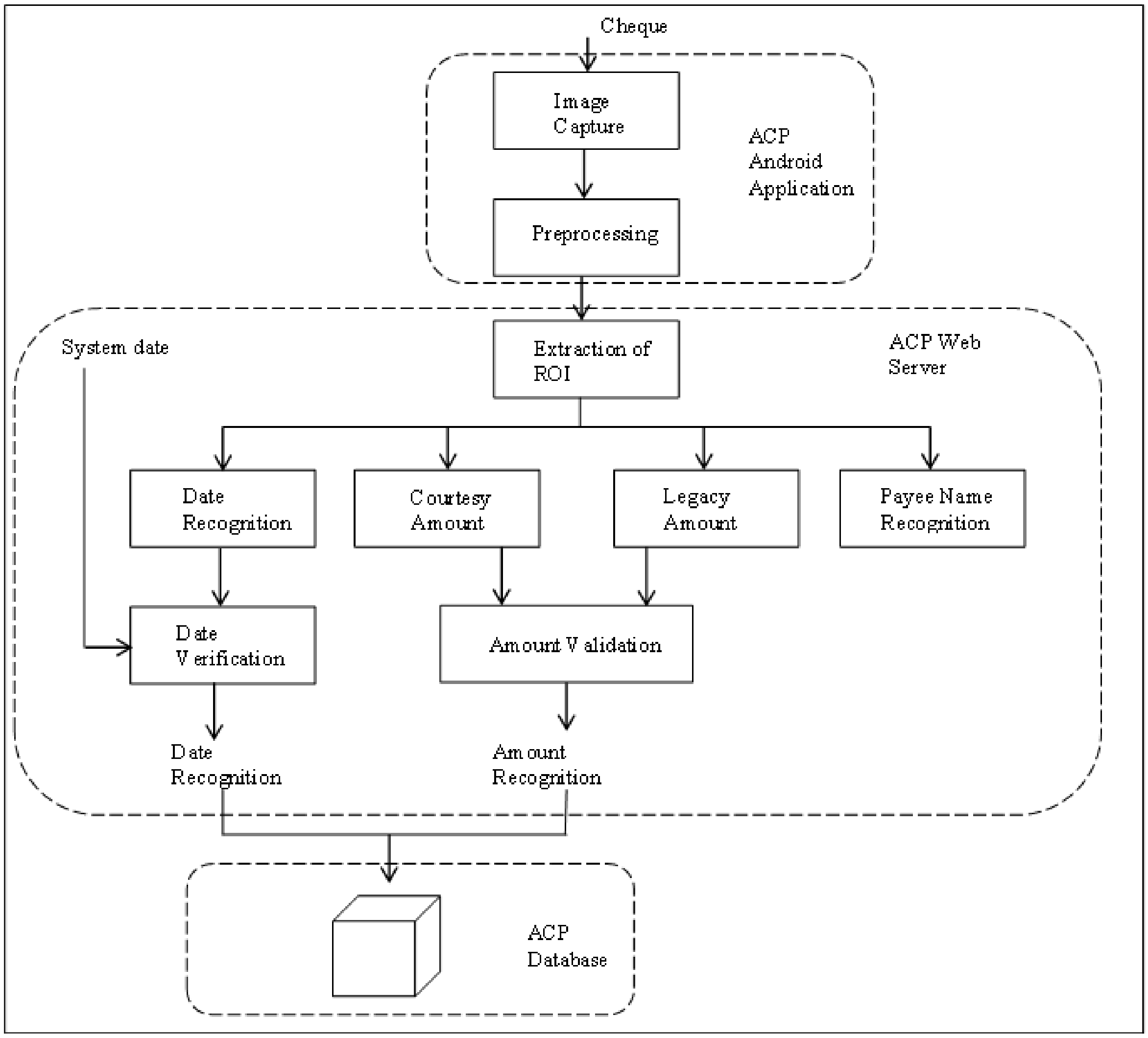


Fig. 2. System Architecture

II. ACP System Architecture

The architectural view[1] of ACP system is as shown in Fig. 2. The system consists of an android client side, Apache Tomcat Web Server and Oracle database.

A. ACP Client

The user must login for using the ACP android application. After login, the user should capture the front side of the cheque which is to be processed. Resolution of the image will be checked for minimum 300 dpi, if not the user will be prompted to recapture. Then the user should upload the image to the server. Server will return the data from the cheque like date, payee name and amount to the android client.

The user will be asked to confirm these details. Further processing will depend on user's confirmation.

B. *ACP Server*

The image uploaded by the user will be stored. In order to increase the accuracy of the OCR the quality of image should be improved by preprocessing. Preprocessing includes converting the image into black and white as well as adjusting the brightness and contrast of the image. OCR will be performed on this image by Tesseract and the data from the cheque will be sent back to the ACP client. If the user chooses to proceed with transaction, the cheque will be approved for further processing.

C. *ACP Database*

The database will contain account details of all users. Any transaction will update the database. All the transaction details will be maintained for each user.

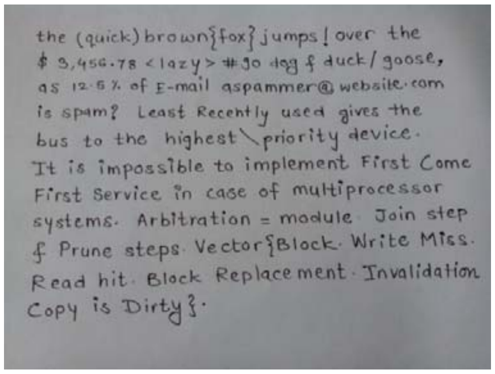
III. Image Preprocessing

In ACP system, the image is captured using android device's camera. The captured image suffers from number of limitations like geometrical distortions, insufficient brightness, folded or torn cheque corners and edges, skewed image[1] etc. Thus need for preprocessing arises. In preprocessing phase, we convert the image into gray scale; adjust its brightness and contrast. We apply different filters to reduce noise from the image. This image is sent to Tesseract for optical character recognition. The preprocessed image increases the accuracy of Tesseract noticeably.

IV. Training Tesseract

Tesseract is an OCR engine developed by HP. It is the most widely used open source OCR tool. Tesseract is better than other OCR engines like GOCR[4] in that, it gives better accuracy and precision, it is highly portable, supports many languages.

Tesseract can be trained[3] to satisfy user's requirements. In ACP we are training Tesseract for recognizing English handwritten text. The first step is to determine the full character set to



the (quick) brown{fox} jumps! over the
$ 3,456.78 <lazy> #90 dog & duck/goose,
as 12.5% of E-mail aspammer@website.com
is spam? Least Recently used gives the
bus to the highest\priority device.
It is impossible to implement First Come
First Service in case of multiprocessor
systems. Arbitration = module. Join step
& Prune steps. Vector{Block. Write Miss.
Read hit. Block Replacement. Invalidation
Copy is Dirty}.

be used and creating .tiff images of handwritten text. Fig. 3 shows the sample image.

Fig. 3. Handwritten text sample image

These images will be used for training with the help of following commands:

A. *Make Box Files*

Tesseract needs a 'box' file to go with each training image. The box file is a text file that lists the characters in the training image, in order, one per line, with the coordinates of the bounding box around the image.

tesseract [lang].[fontname].exp[num].tif [lang].[fontname].exp[num] batch.nochop makebox

e.g. tesseract eng.font1.exp0.tif eng.font1.exp0 batch.nochop makebox

B. *Run Tesseract for training*

For each of your training image, boxfile pairs, run Tesseract in training mode:

tesseract [lang].[fontname].exp[num].tif [lang].[fontname].exp[num] box.train

C. *Compute the Character set*

Tesseract needs to know the set of possible characters it can output. Tesseract needs to have access to character properties *isalpha*, *isdigit*, *isupper*, *islower*, *ispunctuation*. This data must be encoded in the unicharset data file. Each line of this file corresponds to one character. The character in UTF-8 is followed by a hexadecimal number representing a binary mask that encodes the properties. Each bit corresponds to a property. If the bit is set to 1, it means that the property is true. The bit ordering is (from least significant bit to most significant bit): *isalpha*, *islower*, *isupper*, *isdigit*.

unicharset_extractor [lang].[fontname].exp[0].box [lang].[fontname].exp[1].box…

D. Set font_properties

The purpose of this file is to provide font style information that will appear in the output when the font is recognized.

Echo [fontname] [properties] > font_properties

where <fontname> is a string naming the font and <italic>, <bold>,<fixed>, <serif> and <fraktur> are all simple 0 or 1 flags indicating whether the font has the named property.

E. Clustering

When the character features of all the training pages have been extracted, we need to cluster them to create the prototypes.

1. *shapeclustering -F font_properties -U unicharset lang.fontname.exp0.tr lang.fontname.exp1.tr ...*

2. *mftraining -F font_properties -U unicharset -O lang.unicharset lang.fontname.exp0.tr lang.fontname.exp1.tr ...*

3. *cntraining lang.fontname.exp0.tr lang.fontname.exp1.tr ...*

F. Putting it all together

Collect together all (shapetable, normproto, inttemp, pffmtable) the files and rename them with a lang. prefix, where lang is the 3-letter code for your language.

combine_tessdata lang.

We use these commands for training Tesseract.

V. Some Observations

A cheque amount processing system becomes efficient only when it provides high accuracy. A cheque read incorrectly is very difficult to deal with, in terms of costs and time involved to correct the mistake. In ACP system, we preprocess the image before performing OCR. Also we are training Tesseract with large number of handwriting samples with various font styles, in order to increase the accuracy. Also user can terminate the transaction if the cheque data returned to application after OCR is incorrect.

For amount recognition the courtesy amount and the legacy amount should be compared as it will reduce the error rate. In case of the recognition of courtesy amount in a handwritten cheque, the extraction and recognition of the fractional part is difficult[1] as some writers often include fractional parts on cheques.

The date segmentation and recognition task[1] is the most difficult one as there are many formats to write a date. The user should follow the standard format of date.

ACP system is designed to extract the cheque data region wise. It is considered that these regions are located according to Indian standard cheque format. If the format of the cheque differs from the Indian standard, ACP system will not function correctly.

VI. Future work and Conclusion

The ACP system makes the current cheque clearing process efficient and faster. As the system is available to user 24x7, cheque depositing process is not constrained by working hours of banks.

We aim to improve the accuracy of the ACP system by training Tesseract for much more handwritten samples. The current ACP system does not include signature verification. We look forward to implement automated signature verification as a part of ACP. As of today, ACP system handles only Indian standard cheque format. Furthermore, the ACP system can be generalized for different cheque formats.

VII. References

[1] R. Jayadevan, S. R. Kolhe, P. M. Patil , U. Pal, “Automatic processing of

handwritten bank cheque images :a survey”, Springer-Verlag 2011.

[2] “CTS-2010 Standard for Cheque Forms-Specifications”, Dept of Payment and Settlement Systems, RBI, Central Office, Annexure to Circular DPSS.CO.CHD No.1832/04.07.05/2009-2010, Feb 22, 2010.

[3] https://code.google.com/p/tesseract-ocr/wiki/TrainingTesseract3

[4] Shivani Dhiman, A.J. Singh, “Tesseract Vs GOCR A Comparative Study”, International Journal of Recent Technology and Engg. (IJRTE), Sept 2013.

[5] Ray Smith, “An Overview of Tesseract OCR Engine” , Proc. Ninth Int. Conference on Document Analysis and Recognition (ICDAR), IEEE Computer Society (2007), pp. 629-633.

[6] Sandip Rakshit ,Subhadip Basu, “Development of a multi-user handwritten recognition system using Tesseract open source OCR engine” , Proc. International Conference on C3IT (2009) 240-247.

[7] Wojciech Bieniecki, Szymon Grabowski, Wojciech Rozenberg, “Image Preprocessing for Improving OCR Accuracy”,MEMSTECH’2007,May 23-26,2007,Lviv-Polyana,UKRAINE.