Receipts2Go: The Big World of Small Documents

Bill Janssen
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, California 94304
janssen@parc.com

Eric Saund
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, California 94304
saund@parc.com

Eric Bier
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, California 94304
bier@parc.com

Patricia Wall
Xerox Research Center
Webster
800 Phillips Road
Webster, New York 14580
Patricia.Swenton-Wall@Xerox.com

Mary Ann Sprague
Xerox Research Center
Webster
800 Phillips Road
Webster, New York 14580
MaryAnn.Sprague@Xerox.com

ABSTRACT

The Receipts2Go system is about the world of one-page documents: cash register receipts, book covers, cereal boxes, price tags, train tickets, fire extinguisher tags. In that world, we're exploring techniques for extracting accurate information from documents for which we have no layout descriptions – indeed no initial idea of what the document's genre is – using photos taken with cell phone cameras by users who aren't skilled document capture technicians. This paper outlines the system and reports on some initial results, including the algorithms we've found useful for cleaning up those document images, and the techniques used to extract and organize relevant information from thousands of similar-but-different page layouts.

Categories and Subject Descriptors

H.3.1 [**Information Storage And Retrieval**]: Content Analysis and Indexing—*abstracting methods*

General Terms

Algorithms, Design

Keywords

receipt analysis, image normalization, geometric information analysis

1. INTRODUCTION

Though many information transactions have moved to purely digital forms, various types of paper records persist. Items such as register-tape receipts, parking receipts, train tickets, fire-extinguisher inspection tags, and ATM receipts are all interesting oddities in the ever-more-digital world. While all are different, they share the common form of a small typically single-page layout (some are printed on both sides of the paper) with small pieces of information such as dates and dollar amounts and check-marks arranged on the page.

In addition, while the information on any instance from one of the specific document genres is roughly the same as that on any other instance, that information is arranged in a creative profusion of document layouts. Any specific expected piece of information on the form might occur almost anywhere on the form. Consider, for instance, the problem of analyzing a subway ticket from any English-speaking region of the world. It will typically include a date, a purchase price, the name of the underground system, and a serial number of some sort. However, these items may be scattered almost anywhere on the ticket, and use assorted fonts, sizes, and spacings.

In this paper, we present a system that we are developing to handle documents of this type, and walk through an example using a very common document type, the "register-tape" receipt issued by conventional cash registers. Our system is based on a study using ethnographic methods to understand receipt management in small businesses. [10] Receipts are particularly problematic in the modern world because even though they were originally designed as ephemeral, to be quickly discarded, modern tax regulations have forced them to be kept and curated well beyond their original designed lifespans.

We report on some of our results for cleaning up photos of documents to the point where they can be successfully processed by an OCR system, and describe our three-stage follow-on workflow for successfully extracting the pertinent genre-specific information items from a document without a prior model of the document's layout and formatting. Finally, we conclude with some thoughts about where we plan to extend the system in the future.

2. EXISTING SYSTEMS

Production document processing has developed a number of standard technology approaches over the years [15]. The techniques in our system — image normalization, regular expression processing, and rule-based information distilla-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
DocEng'12, September 4–7, 2012, Paris, France.
Copyright 2012 ACM 978-1-4503-1116-8/12/09 ...$15.00.

tion — are not unique, though the specific image-processing algorithms and rule sets used are.

There are a number of commercial products that advertise capabilities for working with small documents such receipts and/or business cards. NeatReceipts [2], which runs on your PC, extracts entities such as currency amounts and vendor names, and can export your database of receipts to a file in various formats, such as Intuit Interchange Format. ScanDrop [4], lets you scan data to the cloud. ProOnGo [3] has apps for smart phones and can sync information across devices. JotNot Scanner [1] provides an app that captures a receipt and sync it to an application like EverNote for later viewing.

3. THE WORKFLOW

Our projected workflow consists of five parts: image normalization, text extraction, entity extraction, entity understanding, and information dispatch. We've made significant progress on two of these, which we discuss in more detail below.

3.1 Image Normalization

Our target image source is the cell-phone camera. However, these cameras vary widely in focus, resolution, and light-gathering ability. Some have flash capabilities of various kinds; some do well in certain lighting conditions and abysmally in others. Thus we need specialized image processing to convert a photo from such a camera (see Figure 1(a)) to a bitonal image suitable for text extraction. This processing includes orientation correction, deskewing, normalization of lighting differences across the surface of the document, and removal of background artifacts and noise elements, where possible.

We apply two improvements that depend on the sizes of the characters on the image. Since the resolution of input images can vary widely, we need to dynamically determine the likely characters on the page so that their sizes can be measured. We do this by applying a clustering algorithm to the connected components of a simple thresholded binary version of the image, which generates two or more clusters of decreasing size. The largest of these clusters is typically noise – speckles and dust on the image. The second largest is taken as the cluster of character images. We measure the average width and height of the components in this cluster and use that measurement as the average width and height of the characters on the image.

Most deskew algorithms work only on a small range of skew angles, so radical skew angles, like 90 degree rotations, must be handled separately; we refer to this process as "orientation correction". Our OCR system will accept either right-side-up or up-side-down images, so our orientation detection need only recognize "sideways" images and correct them. Our system takes advantage of the fact that typical frequently-occurring characters in English text tend to be taller than they are wide. It uses the extracted connected components which represent characters, and calculates the average aspect ratio of those components. If the average is "wider" than it is "tall", the image is taken to be "sideways", and rotated by 90 degrees.

Our system then measures the skew from horizontal of the document image, using a standard Hough transform approach [5]. We check to see if the average height of character connected components is less than some limit (currently we use 20 pixels), and if so, supersample the image by 2x to increase the average character size. We then deskew the possibly supersampled image, using the skew angle measured previously.

Once we have a properly oriented, deskewed, and possibly supersampled image, we remove the background and attempt to normalize the illumination across the foreground content of the image. This is done by an iterative technique that estimates and interpolates the color distribution of light background across the image, and then applies a combination of high-pass filtering and distance measures on the hue-saturation-brightness values of individual pixels. This algorithm is described in detail in [13] and [12].

This normalized image is then binarized using a simple thresholding technique, thus creating a version suitable for OCR input. We then crop the image to cover the area in which the "character-sized" connected components were found, giving us the image shown in Figure 1(b).

3.2 Entity Extraction

We then run our document image through a commercial OCR system to extract the text. We convert this information from the OCR system's proprietary format to the hOCR format standard for OCR output [8], which allows us to effectively separate the particular OCR system from the rest of our system. Bounding box information is present for each word, using the "bbox" hOCR tag. Character and word confidences are present in the hOCR, using the optional "x_confs" and "x_wconf" tags suggested in [8].

The standard workflow for production document processing starts with a classification step, to identify the correct information extraction techniques for a given document, followed by application of those techniques. However, we needed to have a system that would work with a wide variety of documents, and would function correctly even on instances from subgenres which hadn't been encountered before. In addition, we needed it to work without a model-building step which would require a trained technical operator. So we rely on a generic model of receipts and invoices which is based on text features and certain geometric relationships between those features.

We start by applying context-free regular-expression patterns to identify high-value information fragments, which we call "entities", in the OCR text. These are typically structured pieces of information, such as telephone numbers, dates, URLs, names and addresses, currency amounts, email addresses, and credit-card numbers. We also look for certain key phrases like "TOTAL AMOUNT" or "DATE PAID" which help to disambiguate other entities. These phrases are typed with the kind of related entity it would help to disambiguate; for instance, "DATE PAID" would be a "date" key phrase. These entities are shown in Figure 1(c). Note that we've found a number of amounts and amount keywords, as well as a date, a timestamp, a phone number, and an Internet domain.

To allow for OCR errors, we employ a regular expression engine which supports "fuzzy" matching [6] for finding these entities (and also supports very large regular expressions). This allows us to specify regular expressions which match sub-strings within a specified Levenshtein distance of the expression. In addition, the patterns are written to allow for certain typical OCR errors, such as confusing zero with capital or lowercase O.

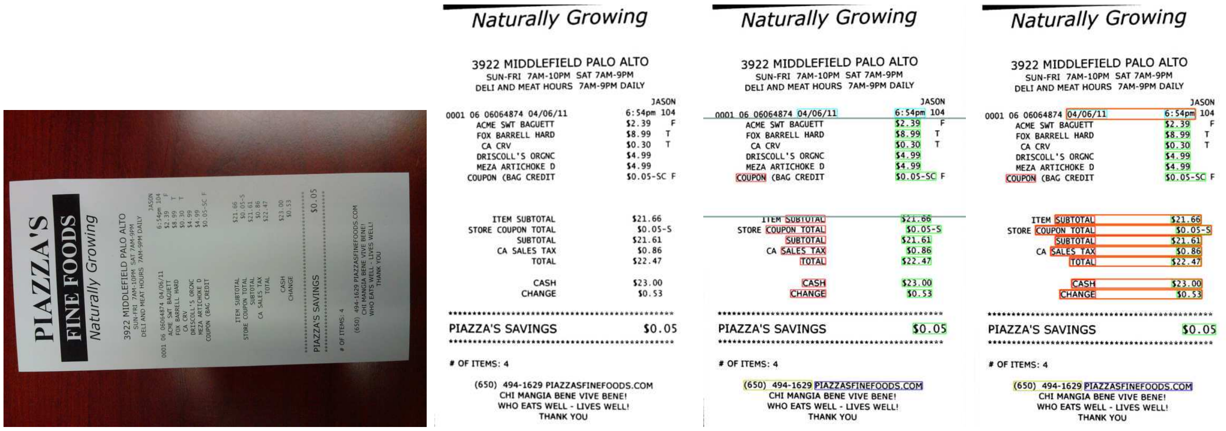

Figure 1: (a) A photo of a grocery store receipt captured by a cell phone. (b) The rotated, deskewed, binarized cropped receipt image ready for OCR. (c) The receipt image, showing the recognized entities and discovered limits of the line-items region. (d) The receipt image, showing various groupings of entity combinations.

We next group each key phrase entity with another entity of the proper type, if the key phrase occurs in a specified geometric relationship with the other entity, with no intervening text. The relationships currently used are "RightOf", "LeftOf", "Below", and "Above". For instance, we use a timestamp as a "date" key phrase entity, and require it to have a "RightOf" relationship with any date it applies to. This helps us to identify the actual date of the receipt, as opposed to an expiration date or special offer date that might also appear on the receipt. Similarly, "amount" key phrases that indicate a total are required to appear "Above" or "LeftOf" a currency amount entity in order to be associated with it. Figure 1(d) shows a number of groupings found in our receipt image.

These groupings, along with the other entities identified, are stored in a file, using an XML format. The bounding box on the image of each entity is stored along with its data and type, and a confidence value; in our current system, the confidence value is simply the minimum of the word OCR confidences for all of the words in the entity. This file is used as input to the rule application step.

To support line item table parsing, we also need to identify the region of the receipt containing the line items. We use a simple algorithm for this, which will work for cash register receipts, but is probably suboptimal for large-form receipts. We first identify the topmost currency amount entity, and use the top of its bounding box as the top of our list items region. We then look for the first currency amount entity below it that is also tagged with an amount keyword, such as "total" or "subtotal". The top of that entity is used as the bottom of the list items region. The left and right sides of the receipt form the left and right sides of the region. Figure 1(c) shows these bounds identified in our receipt image. This information is then stored in the XML file along with the other entity information.

3.3 Information Extraction and User Rule Application

The remainder of our system is rule-based. We use the expert system platform CLIPS [11], a mature, fast, forward-chaining system implemented in C. We apply two distinct stages of rule-based processing to our document. The first stage is designed to either extract or infer additional information about the document; the second is used to apply application-specific rules designed by the user to the extracted information.

Our first stage identifies instances of the two classes we currently support, receipts and invoices. It looks for features — textual keywords and entity patterns — present in a document class and absent in other classes. A classification rule integrates the evidence and comes up with a total score.

Once the document has been recognized as an instance of a particular genre, we apply a set of rules which attempt to extract a uniform set of data appropriate for that genre. For instance, we want to find the total amount a receipt is for, along with the date of the receipt and any tax charged. We do this by examining the entities; e.g., one "receipt" rule locates the currency amount tagged with a "amount" key phrase that contains the word "total" that is lowest on the receipt image, and identifies it as the total of the receipt.

The results of information extraction from our sample receipt are

```
Computed information:
     Date: 4/6/2011
    Total: 22.47
      Tax: 0.86
   Vendor: Piazza's Deli
 Category: 581209 - Delicatessens
```

An important feature of a receipt is the vendor or supplier who provided it. We have rules which take any business names, telephone numbers, or URLs found on the receipt and consult a business directory. This directory contains a list of local businesses, drawn from a nationwide list, which the user can edit or extend. For each business listed, the directory contains its addresses, accounting categories such as "Grocery" or "Hardware", nicknames (often found on receipts), phone numbers, and URLs. If the phone number or URL on the receipt is found to match one of the businesses in the directory within a small edit distance (which varies depending on the length of the text string), that business is assigned as the "vendor" of the receipt. The accounting categories for that vendor are also assigned to the receipt.

In the above example, "Vendor" and "Category" are derived from the store's telephone number.

The "Date" was accepted as the receipt date because of the associated timestamp; receipts which include return information often have multiple dates, including both the date of the original purchase and the date of the return. It's important to distinguish between them, and we've found that the presence of a timestamp is a strong indicator of the correct date.

The second stage of rule-based processing deals with the user-defined rules. They consist of preconditions and resultant "service actions", which are drawn from system-provided application libraries. These rules are stored in an XML-format file which is automatically transformed into additional CLIPS rules, which are then added to the rule base for that user. These rules typically reason over the uniform set of data extracted in the first phase of rule processing, though the full set of extracted entities is also available to user rules.

When a document is detected which matches both a user rule's set of preconditions, and the preconditions for any service actions associated with that rule, the action specified is taken on that receipt. Note that this can be triggered by adding a receipt, but also by changing the preconditions or actions of an existing user rule to match receipts already in the database.

4. FUTURE WORK

There are a number of additional things which could be done here. The vicissitudes of data rates and data plans introduce issues not present in scanner-based systems. They make it desirable to "qualify" a photo before uploading it for further processing. Better image normalization, including image de-warping techniques such as those discussed in [9] and [14], would improve accuracy. Contextual post processing of OCR results to use results from receipts already processed, or from the user's PIM data, might be able to improve the extracted data.

Robust line item identification would enable more varied kinds of actions to be taken. We are currently experimenting with a line-item parser based on a table-parsing algorithm developed by Evgeniy Bart [7]. This uses a machine-learning approach which requires a small amount of explicit labelled data, but has the interesting property of generalizing well to previously unseen tables. Improvement of the feature set for this algorithm, and improvement of the objective function used, are among the things being investigated.

5. CONCLUSIONS

Our Receipts2Go system is capable of automatically extracting pertinent information from a large variety of samples of the receipt and invoice document genres, without having specific format layout models to work from, by using image normalization algorithms, generic entity extraction techniques, and genre-specific grouping rules and and group selection rules. Additionally, it uses the now common cell phone camera as an input device, instead of requiring a specialized scanner. We think these techniques could also be applied to other "small document" genres, such as train tickets or fire extinguisher tags, thus tying those currently isolated pieces of data into the digital world.

6. ACKNOWLEDGEMENTS

Evgeniy Bart contributed his experimental table-parsing framework for our line item identification, and altered his code to make it easier for us to use. Mic Campanelli was kind enough to tweak the FireWorX OCR support in XIPS to allow us to retrieve character and word confidences. MRAB (regex@mrabarnett.plus.com) altered his "regex" library to provide the fuzzy matching we needed. Peter Jarvis, Eric Saund, and Marshall Bern graciously serve as a review and advisory board for this project.

7. REFERENCES

[1] Jotnot scanner. See `http://itunes.apple.com/us/app/jotnot-scanner/id310789464?mt=8`.

[2] Neatreceipts. See `http://www.neat.com/`.

[3] Proongo. See `http://www.proongo.com/`.

[4] Scandrop. See `http://www.officedrop.com/scandrop-scanning-software/scan-google-docs/configure-neatreceipts-scanner`.

[5] J. Anderson, 2008. See `https://github.com/horndude77/image-scripts/blob/master/pnm_java/src/is/image/FindSkew.java`.

[6] M. A. Barnett, 2011. See `http://code.google.com/p/mrab-regex-hg/`.

[7] E. Bart. Parsing tables by probabilistic modeling of perceptual cues. In *Proceedings of the IAPR International Workshop on Document Analysis Systems*, 2012.

[8] T. Breuel. The hOCR microformat for OCR workflow and results. *Document Analysis and Recognition, International Conference on*, 2:1063–1067, 2007.

[9] S. S. Bukhari, F. Shafait, and T. M. Breuel. Coupled snakelet model for curled textline segmentation of camera-captured document images. In *Proceedings of the 2009 10th International Conference on Document Analysis and Recognition*, ICDAR '09, pages 61–65, Washington, DC, USA, 2009. IEEE Computer Society.

[10] B. Janssen, E. Bier, E. Saund, P. Wall, and M. A. Sprague. Receipts2go: Cloud-based automated receipt processing. Technical Report IR_736, Palo Alto Research Center, Inc., December 2011.

[11] G. Riley. CLIPS: An expert system building tool. In *Technology 2001: Proceedings of the Second National Technology Transfer Conference and Exposition*, 1991.

[12] E. Saund. US Patent 7177483: System and method for enhancement of document images, February 2007.

[13] E. Saund, D. Fleet, D. Larner, and J. V. Mahoney. Perceptually-supported image editing of text and graphics. In *UIST 03: Proceedings of the 2003 ACM Symposium on User Interface Software and Technology*, pages 183–192, 2003.

[14] A. Ulges, C. H. Lampert, and T. M. Breuel. Document image dewarping using robust estimation of curled text lines. In *Proceedings of the Eighth International Conference on Document Analysis and Recognition*, ICDAR '05, pages 1001–1005, Washington, DC, USA, 2005. IEEE Computer Society.

[15] P. S. P. Wang and H. Bunke, editors. *Handbook of Character Recognition and Document Image Analysis*. World Scientific Publ. Comp., 1997.