Information extraction by finding repeated structure

Evgeniy Bart
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94394
bart@parc.com

Prateek Sarkar
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94394
psarkar@parc.com

ABSTRACT

Repetition of layout structure is prevalent in document images. In document design, such repetition conveys the underlying logical and functional structure of the data. For example, in invoices, the names, unit prices, quantities and other descriptors of every line item are laid out in a consistent spatial structure. We propose a general method for extracting such repeated structure from documents. After receiving a single example of the structure to be found, the proposed method localizes additional instances of this structure in the same document and in additional documents. A wide variety of perceptually motivated cues (such as alignment and saliency) is used for this purpose. These cues are combined in a probabilistic model, and a novel algorithm for exact inference in this model is proposed and used. We demonstrate that this method can cope with complex instances of repeated structure and generalizes successfully across a wide range of structure variations.

Categories and Subject Descriptors

I.7 [**Computing Methodologies**]: Document and text processing; I.4 [**Computing Methodologies**]: Image processing and computer vision

General Terms

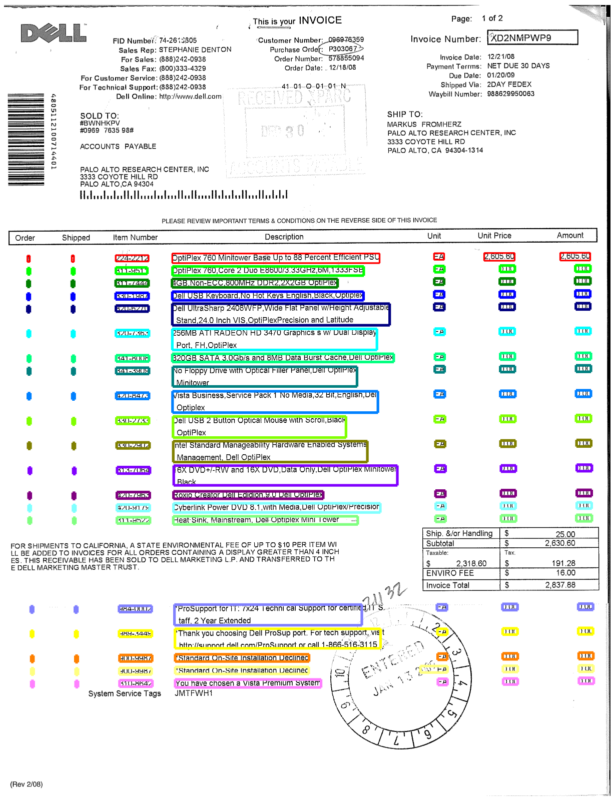
Algorithms

Keywords

information extraction, repeated structure

1. INTRODUCTION

Repeated layout structure in documents conveys vital perceptual and semantic cues to the human reader. Structure can be repeated across entire documents (as in forms, templates, and letterheads), across pages within a single document (as with headings, body text, and captions), and across elements within a single page (as with bulleted lists and tabular data). In each of these cases clues to the human reader



This is your INVOICE

Page: 1 of 2

Invoice Number: XD2NMPWP9

FID Number: 74-2613805
Sales Rep: STEPHANIE DENTON
For Sales: (888)242-0938
Sales Fax: (800)333-4329
For Customer Service: (888)242-0938
For Technical Support: (888)242-0938
Dell Online: http://www.dell.com

Customer Number: 096976359
Purchase Order: P303067
Order Number: 578855094
Order Date: 12/18/08

41 01 O 01 01 N

Invoice Date: 12/21/08
Payment Terrms: NET DUE 30 DAYS
Due Date: 01/20/09
Shipped Via: 2DAY FEDEX
Waybill Number: 988629950063

4805112100714401

SOLD TO:
#BWNHKPV
#0969 7635 98#

ACCOUNTS PAYABLE

PALO ALTO RESEARCH CENTER, INC
3333 COYOTE HILL RD
PALO ALTO,CA 94304

RECEIVED PARC
DEC 30
ACCOUNTS PAYABLE

SHIP TO:
MARKUS FROMHERZ
PALO ALTO RESEARCH CENTER, INC
3333 COYOTE HILL RD
PALO ALTO, CA 94304-1314

PLEASE REVIEW IMPORTANT TERMS & CONDITIONS ON THE REVERSE SIDE OF THIS INVOICE

Order	Shipped	Item Number	Description	Unit	Unit Price	Amount
1	1	224-2212	OptiPlex 760 Minitower Base Up to 88 Percent Efficient PSU	EA	2,605.60	2,605.60
1	1	311-9511	OptiPlex 760,Core 2 Duo E8600/3.33GHz,6M,1333FSB	EA	0.00	0.00
1	1	311-7444	4GB,Non-ECC,800MHz DDR2,2X2GB OptiPlex	EA	0.00	0.00
1	1	330-1987	Dell USB Keyboard,No Hot Keys English,Black,Optiplex	EA	0.00	0.00
1	1	320-6270	Dell UltraSharp 2408WFP,Wide Flat Panel w/Height Adjustable Stand,24.0 Inch VIS,OptiPlexPrecision and Latitude	EA	0.00	0.00
1	1	320-7363	256MB ATI RADEON HD 3470 Graphics s w/ Dual Display Port, FH,OptiPlex	EA	0.00	0.00
1	1	341-8008	320GB SATA 3.0Gb/s and 8MB Data Burst Cache,Dell OptiPlex	EA	0.00	0.00
1	1	341-3909	No Floppy Drive with Optical Filler Panel,Dell OptiPlex Minitower	EA	0.00	0.00
1	1	420-8473	Vista Business,Service Pack 1 No Media,32 Bit,English,Dell Optiplex	EA	0.00	0.00
1	1	330-2733	Dell USB 2 Button Optical Mouse with Scroll,Black OptiPlex	EA	0.00	0.00
1	1	330-2902	Intel Standard Manageability Hardware Enabled Systems Management, Dell OptiPlex	EA	0.00	0.00
1	1	313-7064	16X DVD+/-RW and 16X DVD,Data Only,Dell OptiPlex Minitower Black	EA	0.00	0.00
1	1	420-7963	Roxio Creator Dell Edidion,9.0 Dell OptiPlex	EA	0.00	0.00
1	1	420-9179	Cyberlink Power DVD 8.1,with Media,Dell OptiPlex/Precision	EA	0.00	0.00
1	1	311-9522	Heat Sink, Mainstream, Dell Optiplex Mini Tower	EA	0.00	0.00

FOR SHIPMENTS TO CALIFORNIA, A STATE ENVIRONMENTAL FEE OF UP TO $10 PER ITEM WILL BE ADDED TO INVOICES FOR ALL ORDERS CONTAINING A DISPLAY GREATER THAN 4 INCHES. THIS RECEIVABLE HAS BEEN SOLD TO DELL MARKETING L.P. AND TRANSFERRED TO THE DELL MARKETING MASTER TRUST.

Ship. &/or Handling	$	25.00
Subtotal	$	2,630.60
Taxable: $ 2,318.60	Tax: $	191.28
ENVIRO FEE	$	16.00
Invoice Total	$	2,837.88

211 32

Order	Shipped	Item Number	Description	Unit	Unit Price	Amount
1	1	984-0002	*ProSupport for IT: 7x24 Techni cal Support for certified IT Staff, 2 Year Extended	EA	0.00	0.00
1	1	989-3449	*Thank you choosing Dell ProSup port. For tech support, visit http://support.dell.com/ProSupport or call 1-866-516-3115	EA	0.00	0.00
1	1	900-9987	*Standard On-Site Installation Declined	EA	0.00	0.00
1	1	900-9987	*Standard On-Site Installation Declined	EA	0.00	0.00
1	1	310-8642	You have chosen a Vista Premium System	EA	0.00	0.00

System Service Tags JMTFWH1

ENTERED
JAN 13 2009

(Rev 2/08)

Figure 1: An example invoice with repeated structure extracted. The seven fields of the topmost record (shown in red) were supplied as an example of the structure to be found. The remaining records (other colors) were found automatically. Note that all records are found, even though they are not placed contiguously on the document. Note also that different records occupy different number of text lines. Even though the single annotated record occupies one text line, additional records with both one and two text lines are found successfully.

about the relationships among elements are implicit in the layout structure.

Figure 1 shows an example of an invoice where each record describes a purchased item. The repeated record structure is clearly visible to the human reader and has a meaningful interpretation: corresponding elements of all records contain the same kind of information (e. g., the unit price). Identifying and extracting such repeated structure is useful in a variety of applications. For example, product names extracted from an invoice could be matched to an ERP database to verify receipt before remitting payment. However, despite its recognized value in business workflows, such data extraction tasks suffer from inadequate or unreliable levels of automation and are still largely done manually. The cost of manual data extraction can be quite high; for example, manually processing a single invoice can cost up to 9 Euro [8]. Large businesses may process tens of thousands of invoices per day, leading to high cost of operations.

Automatic extraction of repeated structure from documents is a challenging task. Variations in the contents of individual fields induce significant variability in the structure. This variability includes changes in the field's visual appearance, as well as width and height. These changes, in turn, induce variations in the relative placement of other fields (Figure 4) and in the presence and appearance of field separators. Many cues typically used for data extraction become difficult to exploit in these circumstances. For example, while whitespace gaps form a useful cue for field boundaries, they may be absent in some documents due to overlaps between different fields (interlacing), as in Figures 2 and 4 (see also [11]). If different records occupy different number of text lines, as in Figures 1 and 4, the periodicity structure will be disrupted as well, and the relative positions of different fields will not be consistent across items. All these difficulties make tabular data extraction difficult to approach with shrink-wrapped automated solutions.

In this paper, we propose a semi-automatic method for extracting repeated structure from document images based on one-shot structure learning. In the proposed method, the user provides one example of structure to be extracted. The remaining instances of this structure in the same document (and possibly in additional documents with similar format, e. g. in invoices from the same vendor) are then extracted automatically. The ability to generalize from a single example record makes the process rapidly adaptable and retargetable.

The remainder of this paper is organized as follows. In section 2, we survey the relevant previous work. In section 3, we describe the proposed method for finding repeated structure. The experimental validation of this method is described in section 4. We conclude with general remarks in section 5.

2. A SURVEY OF PREVIOUS WORK

The approach closest to our work is called 'wrapping' [7]. In wrapping, like in our method, one instance of the structure to be extracted is marked by the user. Subsequently, the system 'wraps' (finds and extracts) additional instances of this structure. This wrapping is based on subgraph matching. The biggest drawback of the approach presented in [7] is that the system does not learn from the annotation specified by the user. Instead, the user manually specifies the conditions for subgraphs to match. This requires significant effort and technical expertise. In contrast, our method is completely automatic after a single instance of repeated structure is annotated.

Several methods specifically for extracting information from invoices have been developed. In [6], consecutive lines that share similar token structure are extracted. Similarity is measured by token content (numerical/alphabetic/alphanumeric) as well as by token alignment. In [1], 'main lines' are identified as text lines containing a real number (which usually corresponds to the price). Projection profiles on the main lines are then used to find columns. Periodicity structure of the main lines is used to assign the invoice to one of several predefined types. Similar ideas are used in [9]: a database of known invoice structures is used for invoices of familiar types, and token alignment is used as a cue to identify repeated structure in unfamiliar invoices.

Although some of the methods described above are used in practice, it is desirable to extend their range of applicability. For example, most of the current methods assume fields are organized in widely separated columns, and that no interlacing is present [1, 9]. As a result, documents such as those in Figure 4 cannot be processed. Violation of periodicity (due to varying item height, as in Figure 1) is also difficult to deal with [1, 6, 9]. The main reason for these limitations is that both the cues and the decision functions used to combine these cues are relatively weak. For example, the magnitude of misalignment of different fields is usually thresholded and used as a binary value ('aligned' / 'not aligned'), instead of using the more informative and robust continuous value. The decision function used to integrate the cues is often ad hoc, relying on a series of thresholds to select a set of matches. Using hard thresholds is particularly problematic since, as mentioned above, any single cue might fail for a particular document.

The method proposed here extends the current state-of-the-art by utilizing a wider variety of cues, including many perceptual cues. These cues are integrated in a principled probabilistic framework. As a result, the proposed method can deal with a much wider variety of documents.

3. FINDING REPEATED STRUCTURE

First, we define the terminology used in this paper. Each instance of the repeated structure of interest is called a *record*. For example, in an invoice each record corresponds to a single product or service purchased. Records are composed of individual *fields* (such as 'unit price' or 'quantity'). These fields are laid out in a consistent (but unknown in advance) spatial structure.

Briefly, the proposed system operates as follows. An image of a document containing repeated structure is loaded. After initial preprocessing (such as deskewing and thresholding), the image is presented to the user, who annotates one record of interest. This record is called the *reference record*. The fields of interest in this record (called *reference fields*) are specified by drawing each field's enclosing rectangle. For example, in an invoice, the description of one product can be annotated, as in Figure 1. The task of the system is then to find additional records in the document. Continuing the invoice example, fields that describe other products on the invoice need to be found, as in Figure 1.

This task is accomplished as follows. A set of candidate matches for each reference field is generated. Each possible combination of these individual matches is a candidate record. A probabilistic model is used to evaluate the quality of these candidate records. To avoid explicitly considering all possible candidate records, a novel search algorithm is used. This algorithm is guaranteed to find the globally optimal solution, but examines just a fraction of candidate records, making inference efficient. Since multiple equally good candidate records may be present, we find a redundant set of candidates at this stage. Subsequently, these candidates are filtered, and the final set of records is selected to explain as much of the document as possible. These steps are detailed in the subsequent sections.

3.1 Generating candidate fields

To generate candidate matches for each reference field, a set of tokens is extracted from the document image. The tokens are extracted by OCR (which also gives their text content). These tokens usually correspond to individual words or characters. In contrast, many fields in documents of interest consist of multiple words and can even span multiple text lines. Good matches for a field therefore generally consist of multiple tokens. Therefore, we generate all possible contiguous blocks of tokens and use these as an initial pool of candidates.

In principle, this initial pool could be used as the set of candidate matches for each field. However, it contains many (sometimes tens of thousands) candidates, and considering them all would be time-consuming. Therefore, for each reference field, we discard all candidate fields that are unlikely to participate in a good match with that reference field. This filtering process produces a much shorter list of candidate matches for every reference field. (Note that each reference field will now have a different set of candidates.)

The filtering is performed as follows. We use a probabilistic model to evaluate the likelihood of potential matches. This model is described below in section 3.2. It includes some features that measure how likely a given candidate field B is to match the corresponding reference field R. The values of these features (denoted $f^s(B,R)$) are computed for each candidate match B, and candidates with scores less than a threshold are removed from the candidate list. The thresholds are learned in the training stage, and depend on $f^s(B,R)$, as well as on $f^s(R,R)$ (the corresponding feature value computed for the reference field R itself). The latter dependency is useful for features that measure a field's perceptual coherence. For example, most text fields do not contain large whitespace gaps. A candidate field with such a gap is therefore unlikely to be a good match and can be discarded. But if the reference field itself contains a large gap, then it becomes plausible that a matching candidate field might contain a gap as well. Candidates with large gaps are therefore no longer considered unlikely and need not be removed.

The filtering process described here typically leaves between 20 and 200 candidate matches for each reference field.

3.2 Evaluating a candidate match

Denote the n reference fields by the sequence $\{R_i\}_{i=1}^n$. A candidate matching record consists of n candidate fields (one candidate match for each reference field). The candidate match for field R_i is denoted B_i, so the candidate record is the sequence $\{B_i\}_{i=1}^n$. We propose to evaluate the match quality of a candidate record by modeling the class-conditional probability of the match,

$$p(\{B_i\}_{i=1}^n, \{R_i\}_{i=1}^n | C). \qquad (1)$$

Here the class variable C can take on two values: 'match', or M, and 'no match', or $\overline{M}$. For efficiency, we propose the following specialized form of the model:

$$p(\{B_i\}_{i=1}^n, \{R_i\}_{i=1}^n | C) = a(\{B_i\}_{i=1}^n, \{R_i\}_{i=1}^n | C) \cdot \\ \cdot \prod_{i \in [1 \ldots n]} s(B_i, R_i | C) \cdot \prod_{\substack{i,j \in [1 \ldots n] \\ i \neq j}} d(B_i, B_j, R_i, R_j | C). \qquad (2)$$

Here $s()$ models the probability that an individual field B_i matches its corresponding reference field R_i (s stands for 'single-field criterion'); $d()$ models the probability that a pair of fields B_i, B_j match the corresponding pair R_i, R_j (d stands for 'double-field criterion'); and $a()$ models the overall match quality not modeled by the previous terms (a stands for 'all-field criterion').

The terms $s()$, $d()$, and $a()$ are modeled using the naive Bayes assumption. Features indicating presence or absence of a match are extracted and modeled as conditionally independent given the class. Although the naive Bayes assumption usually does not strictly hold in practice, it often gives good results [2]. More sophisticated methods that can incorporate dependencies (such as [3, 5]) will be explored in the future.

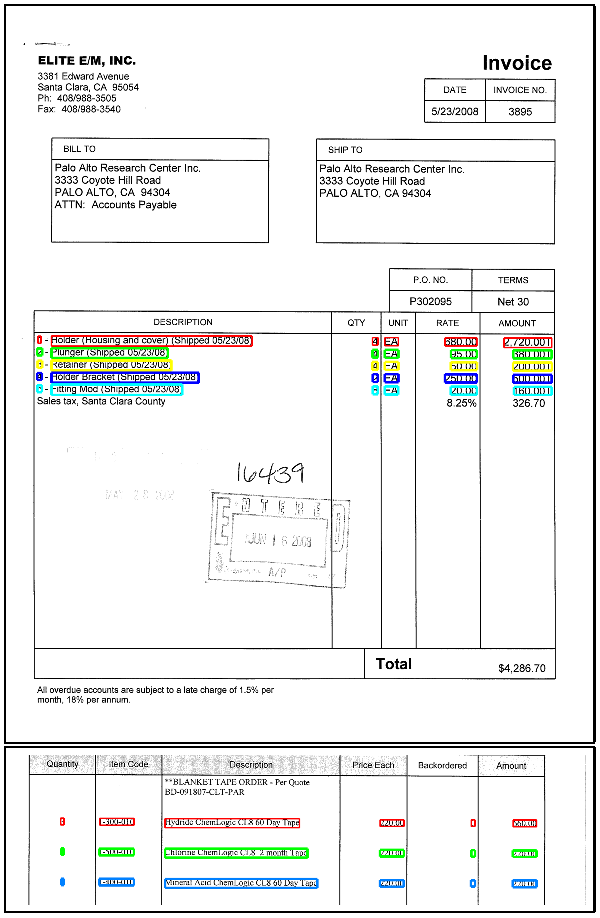
Different kinds of features are extracted for different terms of the model above, according to the purpose of each term.

The $s()$ terms measure the perceptual coherence of candidate fields and assess how well each candidate B_i matches the corresponding reference field R_i. To compute $s(B_i, R_i|C)$, the following features are extracted:

Basic similarity Standard features, such as alignment between R_i and B_i, their height, width, and presence of overlaps, are computed.

Separation Distance from the left edge of B_i to the nearest horizontally overlapping token to the left. The purpose of this feature is to measure how perceptually salient B_i is as a separate entity. If this distance is small, then B_i is not perceptually salient, because it almost blends with another token. Since documents are designed for easy human perception, they are less likely to contain such a field, and its quality as a potential match is therefore reduced. Similarly, features for separation to the right, top, and bottom are computed.

Gaps Size of the largest horizontal gap between neighboring tokens on the same text line. Again, the idea is perceptual coherence. Normally, words are placed relatively close to each other on the same text line; a large gap might suggest that B_i is in fact not a single field, but a union of several fields and is therefore a poor match. If more than one text line is present, the largest vertical gap is also computed.



ELITE E/M, INC.
3381 Edward Avenue
Santa Clara, CA 95054
Ph: 408/988-3505
Fax: 408/988-3540

Invoice

DATE	INVOICE NO.
5/23/2008	3895

BILL TO	SHIP TO
Palo Alto Research Center Inc. 3333 Coyote Hill Road PALO ALTO, CA 94304 ATTN: Accounts Payable	Palo Alto Research Center Inc. 3333 Coyote Hill Road PALO ALTO, CA 94304

P.O. NO.	TERMS
P302095	Net 30

DESCRIPTION	QTY	UNIT	RATE	AMOUNT
1 - Holder (Housing and cover) (Shipped 05/23/08)	4	EA	680.00	2,720.00T
2 - Plunger (Shipped 05/23/08)	4	EA	95.00	380.00T
3 - Retainer (Shipped 05/23/08)	4	EA	50.00	200.00T
4 - Holder Bracket (Shipped 05/23/08)	2	EA	250.00	500.00T
5 - Fitting Mod (Shipped 05/23/08)	8	EA	20.00	160.00T
Sales tax, Santa Clara County			8.25%	326.70
Total				$4,286.70

16439

MAY 28 2008

ENTERED
JUN 16 2008
A/P

All overdue accounts are subject to a late charge of 1.5% per month, 18% per annum.

Quantity	Item Code	Description	Price Each	Backordered	Amount
		**BLANKET TAPE ORDER - Per Quote BD-091807-CLT-PAR			
3	-300-010	Hydride ChemLogic CL8 60 Day Tape	220.00	0	660.00
[illegible]	-500-010	Chlorine ChemLogic CL8 2 month Tape	220.00	0	220.00
[illegible]	-400-010	Mineral Acid ChemLogic CL8 60 Day Tape	220.00	0	220.00

Figure 2: Additional examples of successfully processed invoices. In each case, the fields shown in red were supplied manually as an example of the structure to be found. The remaining records (shown in different colors) were found automatically. To save space, only the relevant portion of the bottom invoice is shown.

Line breaks For each text line except the last, the distance from the right edge of the rightmost token on that line to the right edge of B_i is calculated. The idea here is again to measure coherence of the field B_i. Normally, as much text as possible without overflowing is fitted on each text line, and line breaks are avoided unless the next word does not fit on the current line. Therefore, a large gap suggests that the candidate B_i is not very coherent and may consist of several fields.

Note that most features are motivated by human perception. Since documents are intended for interpretation by humans, it is natural to use such perceptual cues to assess the coherence and match quality of candidate fields. Note also that in addition to the model itself, these features are also used in the filtering process described in section 3.1.

The $d()$ terms are used to determine whether the spatial relations between two candidate fields B_i and B_j match the spatial relations between the corresponding reference fields R_i and R_j. To compute $d(B_i, B_j, R_i, R_j)$, the following features are extracted:

Offsets The difference of the distances between the left edges of B_i and B_j and the left edges of R_i and R_j. The distance between the left edges of two fields measures the horizontal offset of one field with respect to another. This feature therefore measures how different the offsets in the candidate match fields are from the offsets in the reference fields. Similarly, the offsets of the centers and right edges are computed, as well as vertical offsets of top, middle and bottom lines.

Unexplained text Let R be the minimal bounding box that includes R_i and R_j, and B be the minimal bounding box that includes B_i and B_j. If R contains text tokens beyond those included in R_i and R_j, then this feature has value of zero. Otherwise (if all text in R is either in R_i or in R_j), we expect B to contain no additional text except in B_i and B_j. The value of this feature is then the area (in pixels) of such text contained in B, normalized by the square of the average text height in the document.

Finally, $a()$ provides the overall assessment of the match quality between the candidate record and the reference record. The following features are used to compute $a()$:

Intersection A candidate match where some fields intersect is penalized. The area of intersection (in pixels) of the fields' bounding boxes is used for this feature.

Unexplained text The total area (in pixels) of unexplained text in the bounding box B, the minimal bounding box that includes all B_i's. Unexplained text refers to text tokens inside B that do not belong to any B_i.

Note that unexplained text features appear in the $d()$ terms as well as in the $a()$ term. The $a()$ version of this feature directly measures the total amount of unexplained text contained within a candidate record. The $d()$ version is therefore redundant. It is used nevertheless to help optimization, as described in section 3.3.

As mentioned above, the terms $s()$, $d()$, and $a()$ are modeled using the naive Bayes assumption. The log-likelihood ratio therefore can be expressed as

$$\begin{aligned} L(\{B_i\}_{i=1}^n) &= \log \frac{p(\{B_i\}_{i=1}^n, \{R_i\}_{i=1}^n | M)}{p(\{B_i\}_{i=1}^n, \{R_i\}_{i=1}^n | \overline{M})} = \\ &= \sum_{i \in [1 \ldots n]} \sum_k w_k^s [f_k^s(B_i, R_i)] + \\ &\quad + \sum_{\substack{i,j \in [1 \ldots n] \\ i \neq j}} \sum_k w_k^d [f_k^d(B_i, B_j, R_i, R_j)] + \\ &\quad + \sum_k w_k^a [f_k^a(\{B_i\}_{i=1}^n, \{R_i\}_{i=1}^n)], \qquad (3) \end{aligned}$$

where w_k^s is the weight of the $s()$ feature k, and w_k^d and w_k^a are similar weights for the $d()$ and $a()$ features, respectively. These weights are learned from training data, as described below.

3.3 Finding additional records

Given the objective function L in eq. (3), the task is now to find one or several sets of fields $\{B_i\}_{i=1}^n$ that maximize L. This is performed using a search algorithm called best-first leaf search (BFLS) [10]. Briefly, the algorithm uses an upper bound on the objective function and examines candidate solutions in the order of decreasing upper bounds. Once a candidate solution whose objective is above the next-best upper bound is found, the search can be safely terminated since the remaining candidates are guaranteed to have worse upper bounds and therefore also worse objective function values.

The main requirement of BFLS is that the upper bound be factorizable into terms which only involve individual candidate fields. An obvious factorizable upper bound for L in eq. (3) is

$$U(\{B_i\}_{i=1}^n) = \sum_{i\in[1\ldots n]} \sum_k w_k^s[f_k^s(B_i, R_i)]. \tag{4}$$

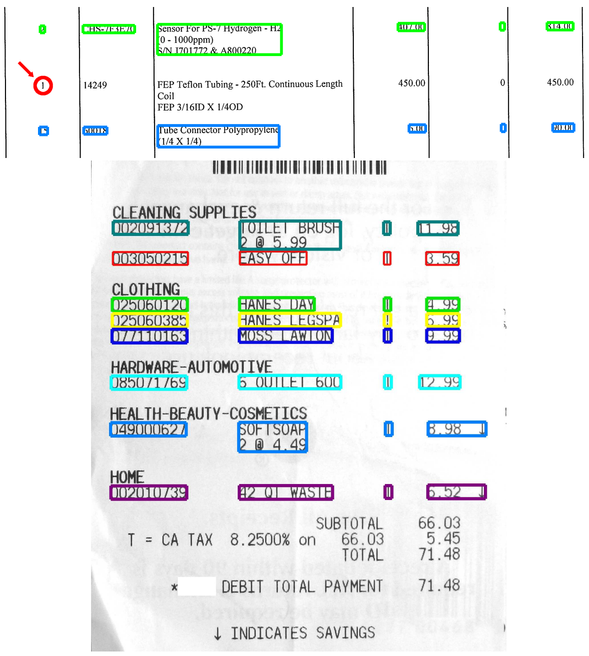
Note that U is a proper upper bound only if all weights w_k^d and w_k^a are non-positive. Most features we use for $d()$ and $a()$ measure discrepancies between an 'ideal' match and the actual candidate match. Large feature values thus correspond to large discrepancies and are therefore expected to be much more common in poorly matching candidates compared to good candidate matches. This implies negative weights for at least large feature values. In practice, some weights may be positive, especially for small feature values. To compensate for that, we subtract the largest weight for each feature from all weights for that feature, making all weights non-positive. This is possible since all features have finitely many possible values (100 in our case, see section 3.5). Subtracting the maximum weight from each feature only changes L by a constant, which is irrelevant since we are only interested in finding the maximum.

The upper bound in eq. (4) is not very tight. The reason is that it involves only the factors matching appearance of a single candidate field to a single reference field, using features such as alignment. A typical document contains many candidate fields that are aligned with any given reference field. Most records composed of these fields will not form a coherent match due to, for example, violation of relative locations structure. But since relative locations do not influence the upper bound, many poor candidates will have a promising (high) value of U and therefore will be considered by the algorithm. This leads to inefficient search.

To overcome this inefficiency, we apply BFLS as follows. First, assume that a match for one of the reference fields (say R_p) is fixed to be B_p. This reference field will be called a 'pivot' below. A new upper bound U^p is

$$\begin{aligned} U^p(\{B_i\}_{i=1}^n) &= \sum_{i\neq p}\sum_k w_k^s[f_k^s(B_i,R_i)] \\ &+ \sum_{i\neq p}\sum_k w_k^d[f_k^d(B_i,B_p,R_i,R_p)] \\ &+ \sum_{j\neq p}\sum_k w_k^d[f_k^d(B_p,B_j,R_p,R_j)]. \end{aligned} \tag{5}$$

Since B_p is fixed, each of the w_k^d terms depends only on one of the candidate fields. Therefore, this upper bound is factorizable. U^p is tighter than U, since U^p includes terms that measure the quality of the spatial relations involving the pivot field p. It is therefore expected that the search problem with the pivot field held fixed can be solved more efficiently. Note also that since U^p involves the $d()$ terms, the solutions which contain unexplained text will be penalized by it, even though the $a()$ terms are not included in U^p. This also improves the efficiency of search.



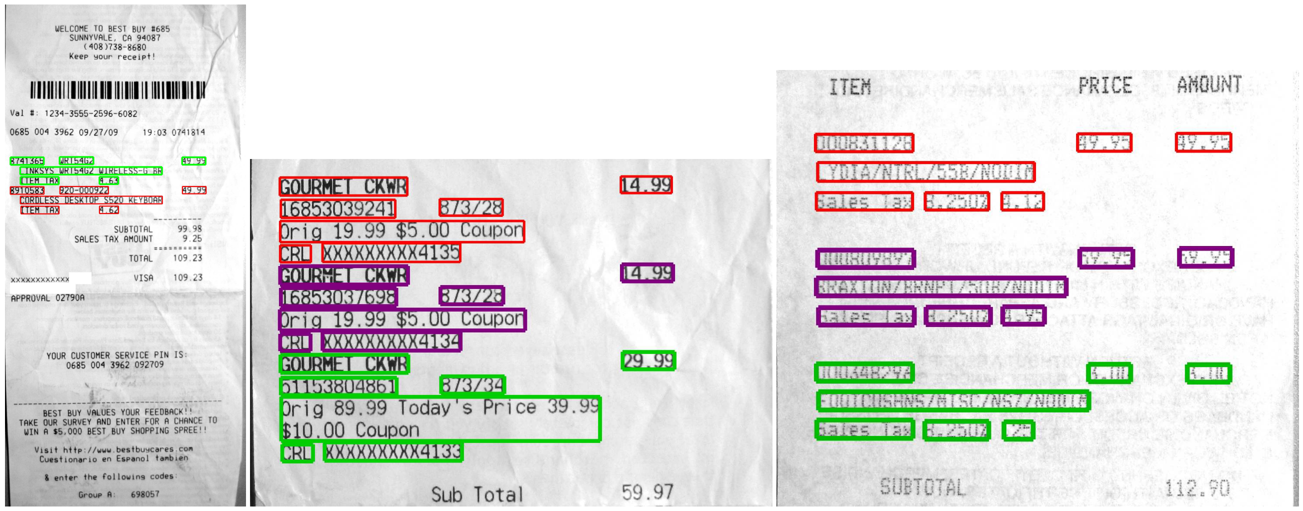
2	CHS-7E3E70	Sensor For PS-7 Hydrogen - H2 (0 - 1000ppm) S/N J701772 & A800220	407.00	0	814.00
1	14249	FEP Teflon Tubing - 250Ft. Continuous Length Coil FEP 3/16ID X 1/4OD	450.00	0	450.00
5	60018	Tube Connector Polypropylene (1/4 X 1/4)	6.00	0	30.00

```
CLEANING SUPPLIES
002091372        TOILET BRUSH     0    11.98
                 2 @ 5.99
003050215        EASY OFF         0     3.59

CLOTHING
025060120        HANES DAY        0     4.99
025060385        HANES LEGSPA     !     5.99
077110163        MOSS LAWTON      0     9.99

HARDWARE-AUTOMOTIVE
085071769        6 OUTLET 600     0    12.99

HEALTH-BEAUTY-COSMETICS
049000627        SOFTSOAP         0     8.98 ↓
                 2 @ 4.49

HOME
002010739        42 QT WASTE      0     6.52 ↓

                         SUBTOTAL      66.03
  T = CA TAX  8.2500% on    66.03       5.45
                            TOTAL      71.48

      *       DEBIT TOTAL PAYMENT      71.48

          ↓ INDICATES SAVINGS
```

Figure 3: Examples of errors in processing. Top: an OCR error caused a record to be missed. The OCR system omitted one token (the text field '1', marked by the red circle and arrow). As a result, the record to which this token belongs could not be identified. The remaining records were found successfully. Bottom: a model error. Several items in this receipt had a discount indicator next to them (a downward-pointing arrow). Trying to explain as many tokens as possible, the model erroneously included these as part of the 'price' field. To save space, only fragments of the original documents are shown.

Therefore, the following modified search algorithm is proposed. One of the fields is selected as pivot. Each candidate match for that field is considered in turn as a possible fixed match. For that fixed match, the reduced problem is constructed and optimized using BFLS with the modified upper bound U^p. This is repeated for each candidate match to the pivot field. Since an optimization problem needs to be solved for every candidate match to the pivot, it is natural to select as pivot the reference field with the fewest match candidates. This modification of BFLS is called pBFLS, for 'pivoted BFLS'.

Using pBFLS, we find $N = 10$ best solutions for each candidate pivot. If there are P pivot candidates, the final set of candidate records has size NP.



WELCOME TO BEST BUY #685
SUNNYVALE, CA 94087
(408)738-8680
Keep your receipt!

Val #: 1234-3555-2596-6082

0685 004 3962 09/27/09 19:03 0741814

8741365 WRT54G2 49.99
LINKSYS WRT54G2 WIRELESS-G BR
ITEM TAX 4.63
8910583 920-000922 49.99
CORDLESS DESKTOP S520 KEYBOAR
ITEM TAX 4.62

SUBTOTAL 99.98
SALES TAX AMOUNT 9.25
TOTAL 109.23

XXXXXXXXXXXX VISA 109.23

APPROVAL 02790A

YOUR CUSTOMER SERVICE PIN IS:
0685 004 3962 092709

BEST BUY VALUES YOUR FEEDBACK!!
TAKE OUR SURVEY AND ENTER FOR A CHANCE TO
WIN A $5,000 BEST BUY SHOPPING SPREE!!

Visit http://www.bestbuycares.com
Cuestionario en Espanol tambien

& enter the following codes:

Group A: 698057

GOURMET CKWR 14.99
16853039241 873/28
Orig 19.99 $5.00 Coupon
CRL XXXXXXXXX4135
GOURMET CKWR 14.99
16853037698 873/28
Orig 19.99 $5.00 Coupon
CRL XXXXXXXXX4134
GOURMET CKWR 29.99
61153804861 873/34
Orig 89.99 Today's Price 39.99
$10.00 Coupon
CRL XXXXXXXXX4133

Sub Total 59.97

ITEM PRICE AMOUNT

000831128 49.95 49.95
LYDIA/NTRL/55B/N001M
Sales Tax 8.250% 4.12

000809897 59.95 59.95
BRAXTON/BRNPT/50B/N001M
Sales Tax 8.250% 4.95

000348298 3.00 3.00
FOOTCUSHNS/MISC/NS7/N001M
Sales Tax 8.250% .25

SUBTOTAL 112.90

Figure 4: Examples of successfully processed receipts. To save space, only the leftmost receipt is shown in full; for the remaining two receipts, only the relevant portion is shown. In each case, the fields shown in red were supplied as an example of the structure to be found. The remaining records (shown in different colors) were found automatically. Note the interlacing present due to each item being described on multiple text lines. For example, in the leftmost receipt, the product description overlaps with all other fields except price, destroying columnar structure. Note also that some fields occupy a variable number of text lines, leading to variations in field layout. For example, in the middle invoice, the 'discounts' field may occupy one line (top two items) or two lines (last item). As a result, the distance between the 'item code' field and the 'CRL' field may be one text line or two text lines. Note that even though in the single annotated record this distance was one text line, additional records with distance of both one and two text lines were found successfully.

3.4 Final filtering

The search algorithm outlined above is used to find a large set of promising candidate records. The next task is to select the best subset of these records. We propose to select a set of solutions that together explain as much of the document as possible. This means that as many tokens as possible must belong to one of the records. The final set also must not have any intersections (i. e., each field must belong to only one record). This selection is achieved by brute-force search and is described next.

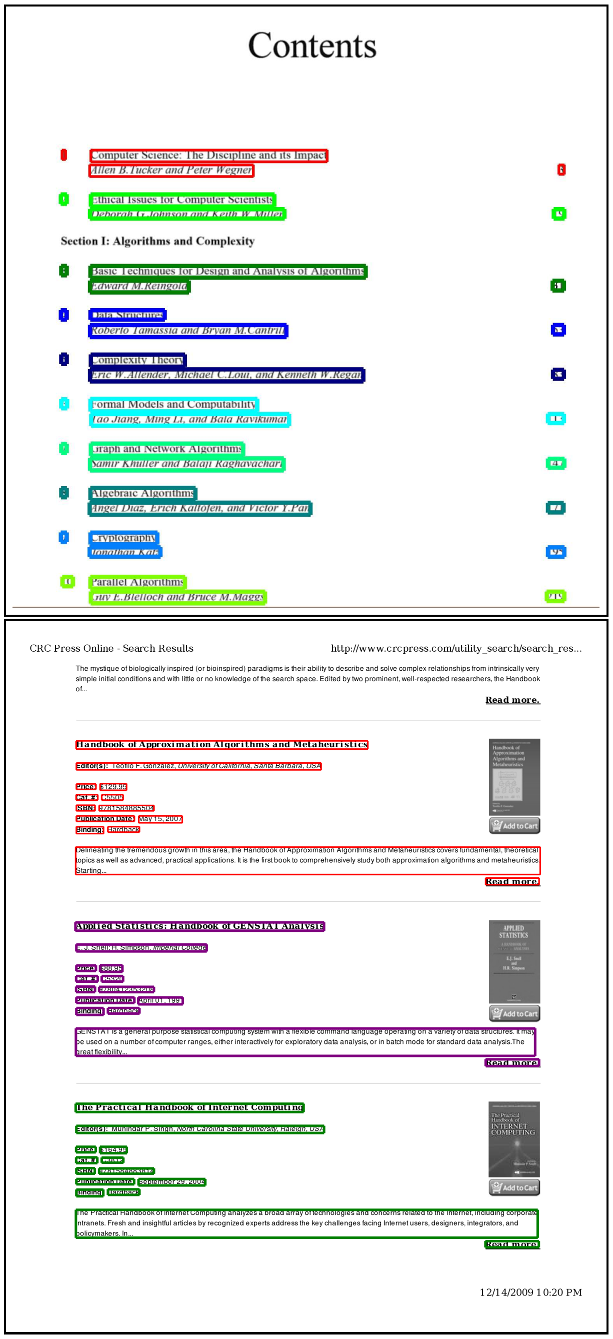
Recall that our set of candidate solutions contains multiple candidates for each pivot value. Obviously, only one solution per pivot can be selected (two solutions with the same pivot value will intersect at least at that pivot). It is therefore sufficient to consider all possible solution sets that consist of one candidate record for each pivot. These solution sets are explored by direct enumeration. Sets which contain intersections are discarded, and the set which explains the most of the document is selected as the final solution set. To compute how much text is explained by a particular solution set, we compute the area (in pixels) of all the text tokens which do not belong to any field of any record in the solution set. This text is called 'unexplained', and we select the solution set which minimizes the area of unexplained text.

The number of possible solution sets that need to be examined is the product of the numbers of candidates per pivot. This number can become prohibitive, especially for large documents. Therefore, it is desirable to filter the solutions to retain only the most promising candidates.

A natural way to perform this filtering is by setting a threshold on the match quality, measured by the log-likelihood function L (eq. (3)). However, the number of terms in L depends on the number of reference fields. The range of variation of L is therefore different for problems with different numbers of reference fields: what is considered a high (good) value of L for a problem with ten reference fields may be a low (poor) value for a problem with only two reference fields. This is because for fewer reference fields, fewer variables are present and therefore fewer overall penalties are incurred for poorly matching variables. As a result, selecting a single problem-independent threshold is difficult.

To alleviate this problem, we propose a modification of L, called L'. The idea is to include some of the information available in L, but to make the number of terms in L' independent of the number of reference fields. To achieve this, note that L involves three types of terms: the $f^s()$ terms (those involving a sinlge B_i), the $f^d()$ terms (those involving two B_i's), and the $f^a()$ terms (involving all B_i's). Consider, for example, the $f^d()$ terms. There are roughly n^2 combinations of two fields (recall that n is the number of reference fields); therefore, there are roughly n^2 terms per feature. These terms provide information about how similar the spatial relations among the candidate match fields are to the spatial relations of the reference fields. Denote the vector of all values of a feature f^d_k (for all pairs B_i, B_j, $i \neq j$) by F^d_k. In L, all values in this vector are used, and since the length of F^d_k depends on n, so does L. For L', we therefore propose to characterize F^d_k by a fixed number of values.

We use five evenly spaced percentile values to characterize



Contents

[illegible] Computer Science: The Discipline and its Impact
Allen B.Tucker and Peter Wegner [illegible]

[illegible] Ethical Issues for Computer Scientists
Deborah G.Johnson and Keith W.Miller [illegible]

Section I: Algorithms and Complexity

[illegible] Basic Techniques for Design and Analysis of Algorithms
Edward M.Reingold [illegible]

[illegible] Data Structures
Roberto Tamassia and Bryan M.Cantrill [illegible]

[illegible] Complexity Theory
Eric W.Allender, Michael C.Loui, and Kenneth W.Regan [illegible]

[illegible] Formal Models and Computability
Tao Jiang, Ming Li, and Bala Ravikumar [illegible]

[illegible] Graph and Network Algorithms
Samir Khuller and Balaji Raghavachari [illegible]

[illegible] Algebraic Algorithms
Angel Diaz, Erich Kaltofen, and Victor Y.Pan [illegible]

[illegible] Cryptography
Jonathan Katz 95

[illegible] Parallel Algorithms
Guy E.Blelloch and Bruce M.Maggs 219

CRC Press Online - Search Results http://www.crcpress.com/utility_search/search_res...

The mystique of biologically inspired (or bioinspired) paradigms is their ability to describe and solve complex relationships from intrinsically very simple initial conditions and with little or no knowledge of the search space. Edited by two prominent, well-respected researchers, the Handbook of...

Read more.

Handbook of Approximation Algorithms and Metaheuristics

Editor(s): Teofilo F. Gonzalez, *University of California, Santa Barbara, USA*

Price: $129.95
Cat. #: C5505
ISBN: 9781584885504
Publication Date: May 15, 2007
Binding: Hardback

Delineating the tremendous growth in this area, the Handbook of Approximation Algorithms and Metaheuristics covers fundamental, theoretical topics as well as advanced, practical applications. It is the first book to comprehensively study both approximation algorithms and metaheuristics. Starting...

Read more.

Applied Statistics: Handbook of GENSTAT Analysis

E. J. Snell; H. Simpson, *Imperial College*

Price: $88.95
Cat. #: C5320
ISBN: 9780412353208
Publication Date: April 01, 199
Binding: Hardback

GENSTAT is a general purpose statistical computing system with a flexible command language operating on a variety of data structures. It may be used on a number of computer ranges, either interactively for exploratory data analysis, or in batch mode for standard data analysis.The great flexibility...

Read more.

The Practical Handbook of Internet Computing

Editor(s): Munindar P. Singh, *North Carolina State University, Raleigh, USA*

Price: $164.95
Cat. #: C3812
ISBN: 9781584883814
Publication Date: September 29, 2004
Binding: Hardback

The Practical Handbook of Internet Computing analyzes a broad array of technologies and concerns related to the Internet, including corporate intranets. Fresh and insightful articles by recognized experts address the key challenges facing Internet users, designers, integrators, and policymakers. In...

Read more.

12/14/2009 10:20 PM

Figure 5: Additional examples of successfully extracted repeated structure. Top: a table of contents. Bottom: a search query result.

F^d_k. Namely, the minimum, maximum, median, 25'th percentile and the 75'th percentile values are used. These values provide rough information about how many spatial relations match well and how many match poorly, but without specific details about how well each particular spatial relation is matched. These features therefore provide less information than the full vector F^d_k. On the other hand, the number of these values is independent of n, and therefore they can be used consistently regardless of how many reference fields there are. Again, the naive Bayes assumption is used to compute a modified log-likelihood ratio $L'(\{B_i\}^n_{i=1})$ based on these percentile features.

A final detail is the following. The naive Bayes assumption is that the features are independent given the class value. This assumption is clearly violated for the percentiles of F^a_k. The reason is that F^a_k has length one, since there is only one way to choose n fields out of n. Therefore, all percentiles have the same value, and clearly aren't independent. Therefore, in practice we only use one percentile value for F^a_k.

Of course, the percentiles of F^s_k and F^d_k are not independent either; for example, the maximum is never smaller than the minimum. Therefore, it is desirable to use a learning method which can cope with dependent features (such as SVM) to perform this final filtering. This is a subject of future work.

Here, we perform final filtering by computing the value of L' for each candidate record and remove the candidates whose L' value is less than a pre-specified threshold.

3.5 Training

The goal of training is to learn the parameters of the models specified above. Specifically, the weights w used in L and L' need to be learned. The standard learning procedure for naive Bayes models is used and is described briefly here.

The input to the learning stage is a training set of documents. In each document, at least two records need to be annotated. (Typically, all records would be annotated, but this is not required.) For each record, bounding boxes around all relevant fields are given.

The weight w for a feature f is learned as follows. The values of f are discretized and clipped at 0 and 99. For each discretized value f_0, the probabilities $p(f = f_0|M)$ and $p(f = f_0|\overline{M})$ are estimated from the training data. The weight is then computed as $w[f_0] = \log \frac{p(f=f_0|M)}{p(f=f_0|\overline{M})}$.

A final note is that this training process can automatically deal with irrelevant features, by assigning them low weights. It is therefore possible to specify many features without having to verify their usefulness.

4. EXPERIMENTS

The proposed system was trained on 10 synthetic invoice images. After that, it was tested on a dataset of 15 invoice types. This dataset consists of scanned invoices from 15 different vendors with different invoice styles (see Figures 1 and 2 for examples). Two to six invoices for each vendor were used, for a total of 51 invoices. Each invoice contains between 2 and 21 records (each record corresponds to a description of one product). The total number of records in all invoices is 409. The number of fields per record varies between four and 11. One record per invoice was used for annotation. The remaining 358 records were used for testing the system. Of these, 330 records (92%) were found successfully. We declare a match to be successful when all fields of the record are identified correctly. Several examples are shown in Figures 1 and 2. On the level of complete documents, we declare a document to be processed successfully if all records were found successfully. In this dataset, 32 out of 51 invoices (63%) were processed successfully. In a production setting, these invoices would not need any further manual processing. In most other invoices, there was only one or two errors to be corrected.

The most frequent mode of failure is the model's tendency to include spurious text in a match, even at the expense of match quality. An example is shown in Figure 3 (using a different dataset of receipts, described below). The reason for this behavior is the requirement to explain as much of the document as possible, and the strong penalty for unexplained text. This could be alleviated in the future with a more representative training set that includes examples where unexplained text is present.

Some of the remaining errors are attributed to OCR. Most of these occur when OCR fails to recognize a token. Since candidate matches in our method are constructed from tokens found by OCR, such an omission is not recoverable. Moreover, since the proposed method cannot deal with missing fields, the entire record which contains the omitted field will not be found. An example of this is shown in Figure 3. In the future, we plan to deal with this issue by using connected sets of black pixels as an additional source of candidate tokens, and by dealing with missing tokens as in [4, 12].

Two additional experiments were performed to show the generality of the proposed method. In one experiment, a set of photographed consumer receipts was used. In another, a set of scanned blood test result reports was used. No additional training was needed; the system trained on the 10 synthetic invoices generalized well to these additional domains. The performance of the proposed method on these datasets was similar to the performance with invoices. Several examples of successfully processed receipts are shown in Figure 4. No examples of blood test results are shown due to privacy considerations.

5. DISCUSSION

We have presented a general method for finding repeated structure in document images. The main novelty of the proposed approach is in formulating a principled probabilistic framework for extracting such structure. This probabilistic formulation allows to avoid premature or ad-hoc pruning of candidate matches. In addition, the approach we introduce uses a wide variety of perceptually motivated cues, some of which have not been applied to this problem in the past, and a novel optimization algorithm for efficiently searching the space of candidate matches.

The performance of the proposed method was illustrated on examples from three different domains (invoices, receipts, and medical documents). Our method covers a broader range of documents in each of these categories than was possible previously. In addition, our approach adapts well to other repeated structure finding tasks, such as finding repeated structure in tables of contents or in search results (Figure 5).

One interesting feature of the proposed method is that we avoid using column structure as a cue. This allows the system deal with documents where individual columns overlap (leading to interlacing). This is in contrast to most other invoice parsing methods which assume column structure.

The proposed method requires human supervision at two stages: during training, when user-labeled training examples need to be supplied, and during processing, when the user specifies (by annotating an example) the structure to be found. The amount of training data needed by our system is already quite small. In our experiments, a system trained on ten invoice images successfully generalized to novel invoices (from vendors not represented in the training set), as well as to receipts and medical forms. The ability to use human input during processing may in fact be beneficial, as it allows the system to be easily retargetable. For example, if only a subset of fields in a given document is of interest, or if it is desirable to split a field into several sub-fields, this can be easily achieved by specifying a suitable annotation. A potential concern is for large volume processing, when annotating one example per document may become expensive. To reduce the amount of annotation needed for such jobs, we performed an experiment in generalizing an annotation from one invoice to additional invoices from the same vendor. The performance with such generalized annotations was comparable to performance with annotations specifically made for each invoice. This approach saves significant annotation effort, since many companies receive most of their invoices from a small subset of vendors [8]. Reducing the need for human annotation further is the subject of future work. Some promising directions of research include [4, 12].

6. REFERENCES

[1] Y. Belaid and A. Belaid. Morphological tagging approach in document analysis of invoices. In *ICPR*, 2004.

[2] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *Information Theory*, 14(11):462–467, 1968.

[4] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, June 2003.

[5] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.

[6] H. Hamza, Y. Belaïd, and A. Belaïd. Case-based reasoning for invoice analysis and recognition. In *Proceedings of the 7th International conference on Case-Based Reasoning*, 2007.

[7] T. Hassan. User-guided wrapping of pdf documents using graph matching techniques. In *ICDAR*, 2009.

[8] B. Klein, S. Agne, and A. Dengel. Results of a study on invoice-reading systems in Germany. In *DAS*, 2004.

[9] B. Klein, S. Gokkus, T. Kieninger, and A. Dengel. Three approaches to "industrial" table spotting. In *ICDAR*, 2001.

[10] P. Sarkar and E. Saund. Perceptual organization in semantic role labeling. In *SDIUT*, 2005.

[11] P. Sarkar and E. Saund. On the reading of tables of contents. In *DAS*, 2008.

[12] M. Weber, M. Welling, and P. Perona. Towards automatic discovery of object categories. In *CVPR*, 2000.