T-Recs Table Recognition and Validation Approach

Thomas G. Kieninger and Bernd Strieder
German Research Center for Artificial Intelligence (DFKI GmbH)
P.O.Box 2080, 67608 Kaiserslautern, Germany
Phone: +49-631-205-3485, Fax: +49-631-205-3210
E-mail: kieninger@dfki.de, WWW: http://www.dfki.uni-kl.de/~kieni/

Abstract

In this paper we resume the *T-Recs* block segmentation and layout analysis approach for tabular documents before we discuss an additional processing step for the proper recognition of potential tables.

While the *T-Recs* results of the processing steps so far look pretty good on documents like articles which might occasionally contain the one or the other table (amongst regular paragraphs), the identification of tables gets confused by logical objects such as recipient address, date or company specific printing within business letters heads if those layout objects occur in a horizontal (*left-of, right-of*) neighborship.

To increase the precision [1] for correctly recognized tables in the above mentioned domain of business letters, we developed an additional processing step with has the purpose to determine *clusters of cells* that show up more evident features of a table than just the left-of, right-of relation between some blocks.

Introduction

Document structure analysis is emerging as a key technology for enabling the intelligent treatment of information. Page decomposition into areas of text, images and graphics and the recognition of information structures in the document provide the key for efficient transmission and storage, as well as for information archiving and retrieval for document databases or digital libraries as well as for inverse printing and document re-use applications.

Thus, the goals of today's Optical Character Recognition (OCR) systems go far beyond the simple transformation of document images into simple sequences of words but rather concentrate upon detection of structural features. But current systems focus on the identification of structural elements like paragraphs, headers and lists but do not consider tabular structures.

[1]The measures used for the message understanding conferences (MUC) express the ratio between the *correctly delivered answers* and the correct (*recall*) or delivered (*precision*) answers.

This is due to the lack of robust and universally applicable table analysis systems which in turn is caused by the manyfold appearances of tables. Nontheless, the information inherent in the layout of a table is of high importance and needs more attention.

When speaking about table analysis, we realize two different research topics: the first one is the correct determination of textual units (such as cells within tables) on the document image, which we refer to as *segmentation* or *structure recognition.* The second topic is the analysis of geometrically arranged blocks and the controlled aggregation of these logical entities to higher level structures. We call this *layout-* or *structure analysis.* While most approaches focus on either of these topics, the *T-Recs* system (TABLE RECOGNITION SYSTEM) covers both aspects. In addition there is a third research topic dealing with tables which we like to call *table understanding*, e.g. the work of (Hurst 1999).

This document roughly resumes the overall functionality of *T-Recs*, starting with the initial block clustering – the central idea and also mentioning the series of required postprocessing steps. For detailed descriptions of these processing steps we like to refer the reader to (Kieninger 1998) and (Kieninger & Dengel 1998a) (all papers are downloadable at `http://www.dfki.uni-kl.de/~kieni/t_recs/`).

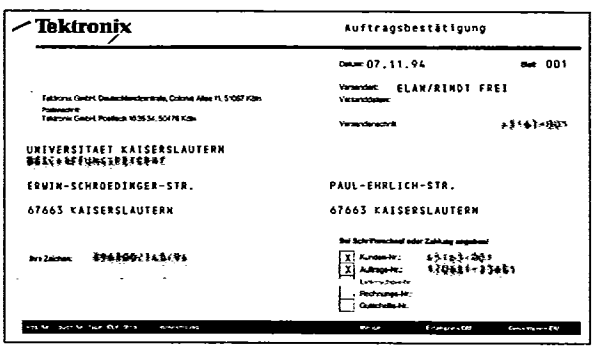

Figure 1: A typical letterhead.

When looking at business letters such as offers, invoices or delivery notes, we typically find the core informa-

tion presented in a tabular way. But not only the cells of these tables are found in a left-of or right-of relation relative to other cells. Also the logical components of the letter head such as recipient address and date or company specific printings are found in a relative horizontal neighborship (see Figure 1). This misleads the simple heuristics of the current *T-Recs* implementation to the assumtption to interpret these entities as compnents of a table.

We addressed this problem before (Kieninger & Dengel 1998b) and tried to apply a preprocessing step with the purpose to focus the *T-Recs* approach to the letter body which in turn contains the tabular core information. But the *Anastasil* system (Dengel 1994) that we applied there is a highly domain dependent approach. Hence, the combination of both systems results in a loss of generality. Consequently, we were looking for a more general approach that allows a more sophisticated determination of block clusters that belong to a table.

The Word—Block Clustering

The investigation of existing table structure recognition approaches discloses a significant similarity: the identification of table elements is always driven by the detection of separators: Rus and Summers(Rus & Summers 1994) present a system for the segmentation and labeling of general structures, also considering table structures where they are able to detect narrow columns using so-called *White Space Density Graphs* (WDG). Other approaches relying on sufficiently large white spaces are described by Rahgozar et al. (Rahgozar, Fan, & Rainero 1994) who operate on word segments rather than on the bitmap and Tupaj et al.(Tupaj, Shi, & Chang 1996) who take plain text output of OCR systems as input.

Others are explicitly looking for ruling lines that determine the table structure. Some representatives here are Green and Krishnamoorthy(Green & Krishnamoorthy 1995), who apply a grammar-based analysis on the set of occuring lines to evaluate the table layout, Itonori(Itonori 1993), who only considers the labeling aspects and expects well segmented blocks as input, or Hirayama(Hirayama 1995) with his interesting *DP matching method.* Chandran and Kasturi (Chandran & Kasturi 1993) consider both (ruled lines and so-called *white streams*) to determine the layout structure.

The core idea of *T-Recs* is to not explicitly look for any kind of separators (lines or spacings) but rather to identify words that belong to the same logical unit. We are not looking for evidences to separate two textual areas but rather look for features that tell us which words belong to one unit and thus build our text blocks from bottom up.

In contrast to other bottom-up segmentation approaches which determine the lines from horizontally adjacent words and the blocks from vertically adjacent lines (like e.g. Gorman(O'Gorman 1992) who has used nearest neighbors to determine wordclusters), our system directly evaluates textblocks based on word segments.

Therefore we take an arbitrary word as the seed for a new block. As seen in the example of Figure 2, we draw a virtual stripe over the bounding box of the word (*consists*) that is to be expanded. This stripe has the width of the bounding box itself and vertically reaches to the directly adjacent lines. We call these words which overlap with that stripe the *horizontally overlapping words* of the block seed and add them to the same block. We visualize this symmetrical relation *ovl*() as *segmentation graph* (seen on the right of Figure 2). The overall block cluster is evaluated as transitive hull *ovl**() of the overlapping relation.

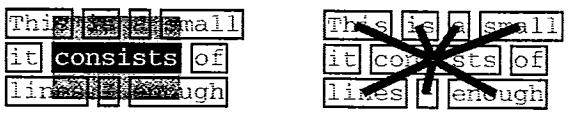

Figure 2: Vertical neighbors of the word "consists".

Strengths of the Approach

The *T-Recs* approach proves its advantages when applied to tabular structures as shown in Figure 3: here we see part of a directory listing with very narrow gaps. Since there are no interleaving words between adjacent columns, the individual column entities are clearly isolated.

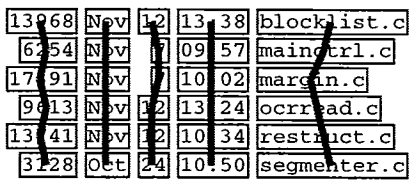

Figure 3: Segmentation of a tabular environment.

Besides this, the algorithm is characterized by the following features:

- It operates on either electronic (ASCII) or paper (OCR output) documents. The necessary word bounding box geometry for ASCII files can easily be derived from a small preprocessor.
- It neglects grid lines. This allows a faster image preprocessing. (The above mentioned ASCII preprocessor detects and suppresses these grid-lines in order to provide comparable input.)
- It disregards textual contents. Thus it can be applied to low quality documents such as faximiles to perform its block segmentation.
- It detects table columns with very narrow gaps (approximately 1 space).
- It detects table-like structures within regular text (e.g. as in logfiles or directory listings) without the presence of characteristic table headers.

- It is universally applicable to any document (no limited, fixed set of table classes; no specific rules and no learning phase necessary)

Inherent Segmentation Errors

The nature of this segmentation algorithm also has some inherent drawbacks. Thus it would not be able to cluster single lines as in headers because they do not have vertical neighbors and thus they would be interpreted as tables with one line height where each word would represent a column. The same problem occurs with words at the end of a non justified block if they do not overlap with the rest of the block.

A second class of errors occurs with regular paragraphs (typically of only a few lines height) that have a space at the same x position on all lines. The paragraph would hence not be identified as only one homogeneous block.

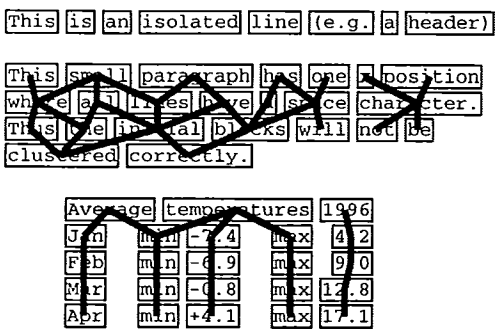

Figure 4: A collection of inherent problems.

A third type of errors occurs in table columns that have a common header. Such a header would by fault glue the underlying columns together. Figure 4 illustrates all these situations together with the initial segmentation graph. To correct these errors a set of postprocessing steps has been developed. Each error class can be identified by some characterisic features and hence goal directed restructuring steps can be applied selectively. For detailed description we like to refer to (Kieninger 1998).

The *T-Recs* Layout Analysis

After having gained the proper segmentation of all elementary blocks (down to table cell level), we further need to exploit these blocks, locate potential table environments and map blocks to their correlating rows and columns. To do this, we reuse a subsystem that was used in one of the postprocessing steps. This module evaluates so called *margin structures* with their *margin points*. The simple occurence of at least two blocks in a horizontal neighborship triggers the creation of such a margin structure.

What we basically do is determine the x-positions of the left and right block bounding box edges for all blocks that are adjacent to such a left-of/right-of block pair. The positions are further marked as of being triggered by a left edge or a right edge and will be called left or *right margin point* accordingly. The set of blocks for which we do this is either terminated by blocks which almost span the whole document width or by sufficiently large vertical white spaces.

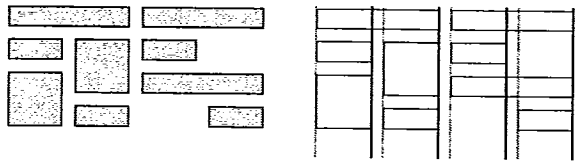

Figure 5: An irregular table and its margin points.

Based on the mapping of these 2-dimensional geometrical information to the 1-dimensional x-axis we can determine the column separtors. Figure 5 gives an example of a irregular table, containing row- and column spanning cells as well as empty cells. The determination of the columns is straightforward: while the areas between a left and a right margin point (looking from left to right) mark the areas for the columns, the transition from a right to a left margin point marks the space for a virtual column separator.

Similarly, the construction of the table row separators is based on the top edges of the block bounding boxes which are mapped to the y-axis. The combination of column and row separators results in a regular 2-dimensional grid structure. We call the areas defined by this grid the *table tiles*. These tiles build the base for the *T-Recs* html output module. For further information about this procedure we like to refer to (Kieninger & Dengel 1998a).

Sample Documents and *T-Recs* Results

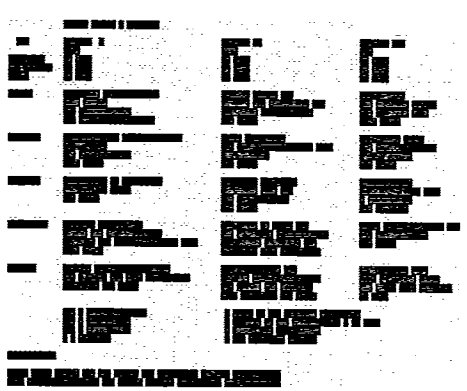

Figure 6: Word segments of sample document.

Let us have a look at some documents and the appropriate results of the *T-Recs* system. Figure 6 shows the the bounding boxes of the word segments – which is the primary input to *T-Recs*. The small blocks are arranged to a table without explicit delineation.

Figure 7 not only visualizes the correctly clustered textblocks, as the wordsegments which belong to the same block are connected with gray rectangles, but also shows a virtual table grid which has been derived from the internal structured representation of the table logic. This constructed grid not only shows the identified table, but also indicates its extension and its rows and columns.

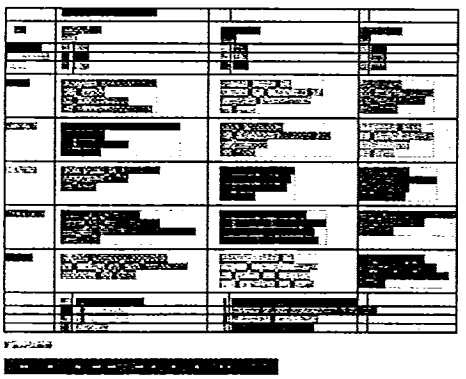

Figure 7: Recognized table structure within sample document (system output).

Figure 8 shows a document which does not consist of a sole table but rather only contains a table which is nested in between regular paragraphs of text. Hence, this example demonstrates the ability of *T-Recs* to deal with mixed-mode documents containing both tables and non-tables. Again we see the input to *T-Recs*, the word segments (left) and the segmented blocks with a constructed table grid (right).

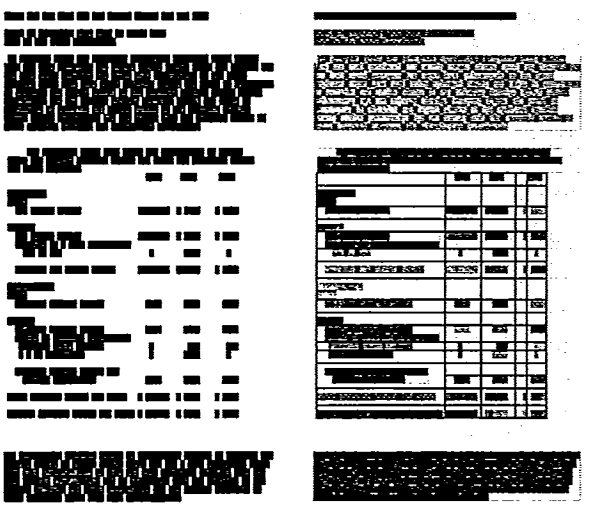

Figure 8: Wordsegments and recognized tables of second sample.

T-Recs applied to Business Letters

As already metioned in the introduction, the neglecting assumption that wherever two or more blocks occur in a left-of/right-of relation we are facing parts of a table (i.e. table cells) leads to misinterpretaions of the logical object in a letter head. Figure 9 shows the bitmap of a sample letter and the areas that *T-Recs* interprets as tables (dark gray areas) [2].

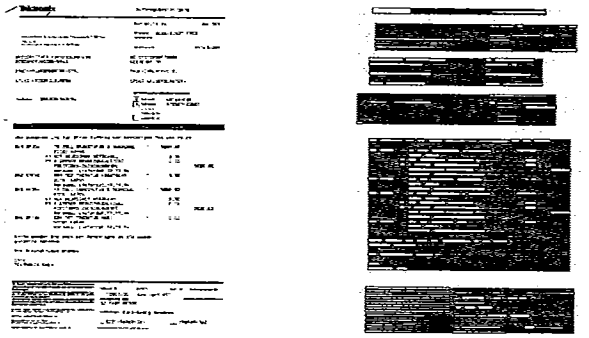

Figure 9: The demoletter and located tables.

It is obvious, that only one of the six assumed tables really contains tabular informations in the usual sense of relational data sets. This is the table located in the letter body. In the context of automated document processing and information retrieval it is not only of importance to locate and analyze occuring tables but rather also to avoid misinterpretations of non-tables. To prevent this potential confusion of *T-Recs*, we need to have a more restrictive heuristics to determine the blocks that belong to a table.

The Block—Table Clustering

To do so, we collected typical features, which can be determined on the block layout itself. They should be as general as neccessary not to loose proper tables and at the same time enough restricive to eliminate the non-tables. The collected features resulted in a binary relation $inclust(b_1, b_2)$ between block objects. The block cluster is evaluated based on the transitive hull $inclust^*()$ of that relation. The $inclust()$ relation makes use of the following features:

Vertical alignment Derived from offset between top and bottom edge of block bounding box.

Horizontal alignment Derived from offset between left and right edge of block bounding box.

Distance Derived from distance between the two blocks (relative to document size).

The determination of these features leads to a single number which can be interpreted as the strength of the

[2] This document shows an extreme number of overdetermined tables. It was selected to emphasize the benefits of the new approach.

relation between two blocks. This value changes continously and monotonously in horizontal and vertical distance and some other values determining left, right or centered alignment. This gives some possibilities to have tunable coefficients for this *inclust*-relation. Two blocks are said to be related when the calculated value lies in a certain range.

The better the alignment, the longer the possible distance for two blocks to stand in that *inclust*-relation. Figure 10 shows two possible constellations for tables. The various connecting edges between blocks indicate the presence of the *inclust*-relation for many block-pairs of both clusters. A caracteristic value could be the ratio of 30 edges to 12 blocks (2.50) for the left table and 20 edges to 9 blocks (2.22) for the right table.

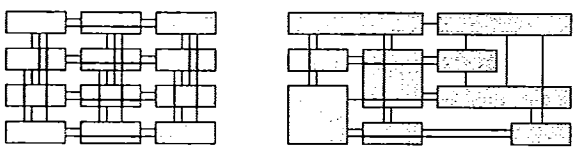

Figure 10: Visualization of the *inclust*-Relation for two tables.

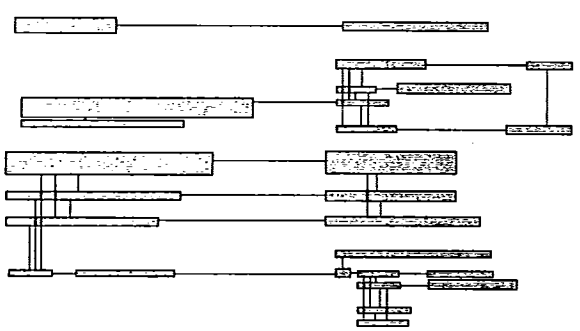

Figure 11: Visualization of the *inclust*-Relation for the sample letter head of Figure 1.

Let us have a look at the head of the sample letter of Figure 1. Figure 11 shows the block segments as determined by *T-Recs* together with the edges indicating the relation which is in this case purely based on alignment (even allowing some tolerances). The corresponding ratio between 38 edges and 27 blocksegments is 1.41. This is significantly less than in the sample tables of Figure 10.

In the constructed samples we moreover weighted all edges identical. No matter how sharp the alignment between two blocks or their relative distance is. The tuning of appropriate parameters could indeed lead to even more significant results.

The transitive hull is not calculated for all blocks at once, but zone by zone. Figure 12 shows a sample document together with its zones. The zones were introduced for two reasons:

1. These zones are selected by clusters of ruling lines (further called *rulers*), vertical spaces and by blocks not belonging to a table which mark the boundaries of the zones. This gives the possibility to use this additional information, which has not been fully used earlier.
2. There was also a need to find zones to restrict the calculation, because the naive calculation of the hull occasionally heavily outweighs all other calculation steps of *T-Recs*.

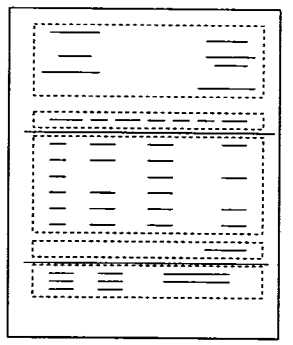

Figure 12: Independent zones of a sample document.

The rulers are clustered to get more evidence. Four types of clusters are distinghuished in *T-Recs* (see Figure 13):

(a) Closed clusters, containing horizontal and vertical rulers, whose convex hull, roughly spoken, is mainly determined by whole rules,
(b) Open clusters, not closed clusters, whose convex hull is mainly determined by ending points of rules.
(c) Vertical clusters consisting of vertical rules inducing a single rule,
(d) Horizontal clusters consisting of horizontal rules inducing a single rule.

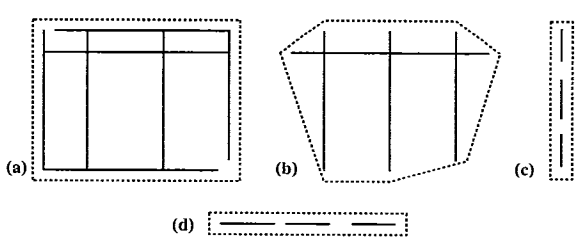

Figure 13: The four types of ruler-clusters.

Limitations of the Approach

This block to table clustering was our first approach towards a more restrictive determination of cells-cluster which actually constitutes a table by calculating a characteristic measure. Based on a given threshhold we

decided whether or not such a cluster should be treated as a table or not. Nonetheless we still faced some problems:

1. How can we determine the proper boundaries for the block clusters? We therefore relied on the zones which themselves were determined by elements such as charcteristic vertical spaces or regular textblocks that are almost as wide as the document. Our approach failed, if no appropriate delimiters were found and hence the region for a table was not identical to any of the zones.
2. How can we decide whether a horizontal ruler (type *d*) should be treated as a separator for either different zones or for different table rows within the same zone.
3. How can we find a universal threshhold value on which to decide whether or not a cell cluster should be treated as a table. In our tests, the tolerances for this value were quite narrow. We did not feel very confident as to find more segregating features for this one-step approach.

The Block—Column—Table Clustering

These limitations forced us to search for an alternate approach. We observed that there are much more similarities in between the cells of one column than between cells of a row. This lead us to a two-step approach where we first determine the columns as aggregations of blocks and then the table itself as aggregation of columns. The similarities between two blocks of the same column are of different nature:

Vertical alignment We determine the deviation of the x-coordinates of the blocks and build measures for *left-*, *right-* and *centered alignment.*

Shape Here we compare the *width* and the *height* (as number of lines) of the blocks and derive appropriate normalized measures. We also consider the different number of words between blocks.

Token structure For columns consisting of single-word blocks (especially for numerical columns) we identified typical structural similarities like number of digits, position of decimal point or same unit names (see Figure 14). Appropriate text matching routines determine a similarity measure.

Text structure For blocks consisting of more than one word, we identified a similarity measure that is strongly related to the *token structure.* Such blocks inside a column tend to have an equal sentence structure with equal words at equal positions and only some "variable" words. Figure 15 (column: "Bezeichung") gives an example.

For the individual shape- and alignment measures we need to deal with different lengths as input and hence operate on pixel coordinates. But these vary with the scanning resolution. Moreover the values need to be put into relation to e.g. the overall document size or

Wir bieten ff. Position(en) freibleibend an:

Pos	Art.-Nr.	Bezeichnung	Menge	Preis DM
1	EA 7402-NLED	Dotm.40x2 5,5LED-STW	5 St.	106.00
2	EA 017-9UKE	Rahm.75x24,2/entspi.	7 St.	15.80
4	EA VK-2128	Graf.128x128+HD61830	1 St.	345.00
5	EA 017-15U	Rahmen 68x68 Grafik	1 St.	13.80

Figure 14: Example for similar token structure.

the fontsize. In each case we are aiming towards independent metrics within the interval $[0, 1]$.

For the evaluation of the token similarity we make use of the Levenshtein distance (Levenshtein 1965). The Levenshtein distance $ld(a, b)$ between two strings a and b denotes the minimum number of edit operations that are needed to transform word a into word b, where the edit operations are either *replacing*, *inserting* or *deleting* of one character.

But for numerical columns we are rather interested in the structural similarity (which might be expressed by e.g. the total number of digits, position of decimal point and commas etc.). Therefore we evaluate a slightly modified function $ld^{alpha}(a, b)$ that treats the digits $(0 \ldots 9)$ as equal characters. Like this, we achive very little distances between the number tokens of typical columns. For the rightmost column of Figure 14 this modification reduces the average value of 3.3 for $ld(a, b)$ to 1.0 for $ld^{alpha}(a, b)$ whereas the distance for alphabetic words remains identical to $ld(a, b)$.

Pos.	Anzahl/ME	Bezeichnung	Art-Nr	Einzelpreis/ DM
1	1 St.	Toshiba Notebook T1910/120, 486SX/33MHz MS-DOS 6.2, Windows 3.11, BallPoint	T01912	2,697.00
2	1 St.	Toshiba Notebook T1910/200, 486SX/33MHz MS-DOS 6.2, Windows 3.11, BallPoint	T01920	2,997.00
3	1 St.	Toshiba Notebook T1910CS/120,486SX/33MHz MS-DOS 6.2, Windows 3.11, BallPoint	T019C12	3,727.00
4	1 St.	Toshiba Notebook T1910CS/200,486SX/33MHz MS-DOS 6.2, Windows 3.11, BallPoint	T019C20	4,157.00

Figure 15: Example for similar text structure.

We stated, that the textual similarity is somehow related to the token similarity, and indeed we make use the Levenshtein distance again – but on a different meta level: While the classical application compares the characters of a string, we compare the words of a text sequence. We use the function $ld^{meta}(a, b)$ which interprets the words of a block as the charcters that might be *replaced*, *inserted* or *deleted* if they are different. The word sequences of two blocks make up the strings that are to be unified.

Again, the values for the token- and text similarity are mapped into the interval $[0, 1]$. This is done by putting them in relation to the maximum length of the strings (or the number of token per block respectively).

For all individual evidences a smaller value stands for higher similarity. But there are differences in the effect of a slight variation of any of the values towards our overall evidence. To take care of these circumstances, all individual values are mapped using a modification of a sigmoid function $\mathcal{S}_c(x) := 1/(1 + e^{-cx})$ which is

typically used in neuronal networks to controll the output of neurons. Our transformation function is defined as:

$$\mathcal{K}_c^{P05}(x) := 1 - \mathcal{S}_c(x - P05)$$

Figure 16 shows examples of this function with different parameters c and $P05$. It is obvious that c influences the steepness while $P05$ determines the x-position for which the function value equals 0.5.

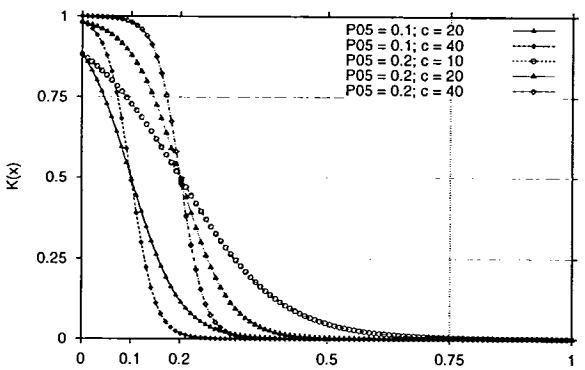

Figure 16: Samples of $\mathcal{K}_c^{P05}(x)$.

For each of these similarity features between two blocks a and b, we conducted am individual $[0,1]$ normalized evidence (with $\Delta funct^{rel}(a,b)$ denoting the appropriate normalized measures):

$$\begin{aligned}
\mathcal{E}_{left}^{c,P05}(a,b) &:= \mathcal{K}_c^{P05}(\Delta xleft^{rel}(a,b)) \\
\mathcal{E}_{right}^{c,P05}(a,b) &:= \mathcal{K}_c^{P05}(\Delta xright^{rel}(a,b)) \\
\mathcal{E}_{cntr}^{c,P05}(a,b) &:= \mathcal{K}_c^{P05}(\Delta xcntr^{rel}(a,b)) \\
\mathcal{E}_{height}^{c,P05}(a,b) &:= \mathcal{K}_c^{P05}(\Delta height^{rel}(a,b)) \\
\mathcal{E}_{width}^{c,P05}(a,b) &:= \mathcal{K}_c^{P05}(\Delta width^{rel}(a,b)) \\
\mathcal{E}_{words}^{c,P05}(a,b) &:= \mathcal{K}_c^{P05}(\Delta words^{rel}(a,b)) \\
\mathcal{E}_{token}^{c,P05}(a,b) &:= \mathcal{K}_c^{P05}(\Delta token^{rel}(a,b)) \\
\mathcal{E}_{text}^{c,P05}(a,b) &:= \mathcal{K}_c^{P05}(\Delta text^{rel}(a,b))
\end{aligned}$$

These evidences together with an individual, positive weight are further aggregated to an overall evidence $\mathcal{E}_{agg}(a,b)$. The higher the individual similarity factors, the higher $\mathcal{E}_{agg}$ and the more confidence for two blocks to belong to one column we have.

$$\begin{aligned}
\mathcal{E}_{agg}(a,b) := \; & \alpha * \mathcal{E}_{left}(a,b) + \beta * \mathcal{E}_{right}(a,b) + \\
& \gamma * \mathcal{E}_{cntr}(a,b) + \delta * \mathcal{E}_{height}(a,b) + \\
& \eta * \mathcal{E}_{width}(a,b) + \lambda * \mathcal{E}_{words}(a,b) + \\
& \xi * \mathcal{E}_{token}(a,b) + \rho * \mathcal{E}_{text}(a,b)
\end{aligned}$$

As the individual evidences are $[0,1]$ normalized, the aggregated evidence ranges from 0 to $\mathcal{E}_{agg}^{max}$ with:

$$\mathcal{E}_{agg}^{max} := \alpha + \beta + \gamma + \delta + \eta + \lambda + \xi + \rho$$

Block—Column Aggregation

We did not use the distance between two blocks to influence $\mathcal{E}_{agg}$ because the spaces between cells of one column might vary and moreover these distances are not typically different to the distances between other layout objects such as regular paragraphs. But we can say that it is more likely that two objects belong to one column if they are closer together. Hence we use the vertical distance $dist_y(a,b)$ to determine our threshhold. To be independent of scanning resolution, we put $dist_y(a,b)$ in relation to the average line height of both blocks a and b. This realtive distance $dist_y^{rel}(a,b)$ is *not* normalized to the interval $[0,1]$. To get a $[0,1]$ normalized threshhold, we use another transformation function which also relies on the sigmoid function $\mathcal{S}_c(x)$:

$$\mathcal{F}_{thresh}^{c,P05}(a,b) \;:=\; \mathcal{S}_c(dist_y^{rel}(a,b) - P05)$$

The predicate $\mathcal{P}_{col}(a,b)$ that we use to cluster the blocks to columns is straightforward: If the weighted aggregated evidence $\mathcal{E}_{agg}(a,b)$ is bigger than the threshhold $\mathcal{F}_{thresh}^{c,P05}(a,b)$ for those blocks multiplied by $\mathcal{E}_{agg}^{max}$, we say they belong to the same column. This is expressed with the predicate $\mathcal{P}_{col}(a,b)$. The columns themselves are constructed as the transitive hull $\mathcal{P}_{col}^*$:

$$\mathcal{P}_{col}(a,b) := \begin{cases} 1 \text{ if } \mathcal{E}_{agg}(a,b) \geq \mathcal{E}_{agg}^{max} * \mathcal{F}_{thresh}^{c,P05}(a,b) \\ 0 \text{ else.} \end{cases}$$

Column—Table Aggregation

Realizing a solution for this final column merging is our current work. While in most cases this merging is a straightforward step, there are still some examples (which we found on business letters) that require some specialized extra investigations:

When putting the horizontally adjacent columns together, we occasionally encounter the problem of having columns ending on different logical table rows. Hence, it is ambiguous which column to take as references to determine the overall table expansion. However, we have to deal with only a small number of different top- and bottom boundaries as defined by each column.

Therefore we favour a "generate and verfiy" approach, where we first generate each possible alternative for the overall table and then evaluate an adequate score for the different table alternatives. The features that we want to consider are:

- Determine the top and bottom table table rows that the most columns have in common.
- The aggregated evidence $\mathcal{E}_{agg}(a,b)$ between pairs of vertically adjacent blocks.
- A column index value that is a transformation of the aggregated evidences $\mathcal{E}_{agg}(a,b)$ to the overall column.
- The vertical alignment of horizontally adjacent cells.

Conclusion and Outlook

While classical character recognition systems do not show recent significant improvements (Rice, Jenkins, & Nartker 1995) (Rice, Jenkins, & Nartker 1996), commercial OCR systems more and more focus on the detection of structural information (such as tables) as key technology for their products. Thus, the *T-Recs* approach pretty much meets the current trends.

The proposed evaluation of table cell clusters based on more complex heuristics leads to an increased recognition precision when applied to tabular business letters such as offers, invoices or delievery notes. In contrast to the approach proposed in (Kieninger & Dengel 1998b) we do this without pruning the applicability to a restricted document domain. We rather use more natural features of tables such as alignment or centering of table cells relative to each other, and the fact that the columns of cells do not have very large white gaps.

Nonetheless, these observations of an improved accuracy are still made on a subjective impression and are not based on objective evaluations. To improve this, it requires some benchmarking technologies for table documents. In this context we have recently developed a frontend to gather ground truth data for given document collections. Like *T-Recs* itself, this module is oriented on the bottom-up document view and hence defines all higher level instances as aggregations of their underlying objects.

Our custom ground truth format that we developed for this module keeps all layout information together with the complete logical markup for structures such as regular paragraphs, ordered and unordered lists, headings and of course tables as well as any arbitrary domain specific markup (e.g. *date*, *address* or *letter body*). Thus, our benchmarking workbench is applicable to other layout analysis tools as well.

An analysis tool to check the differences between the *T-Recs* results and the ground truth data is about to be finished. Its results not only allow us to build objective quality measures and to compare concurrent systems, but most important it allows to automatically evaluate new or alternative processing steps and to tune the system parameters.

References

Chandran, S., and Kasturi, R. 1993. Structural recognition of tabulated data. In *Proc. of International Conference on Document Analysis and Recognition - ICDAR 93.*

Dengel, A. 1994. About the logical partitioning of document images. In *Proceedings SDAIR-94, Int'l Symposium on Document Analysis and Information Retrieval, Las Vegas, NV*, 209–218.

Green, E., and Krishnamoorthy, M. 1995. Recognition of tables using table grammars. In *Proc. of the 4-th Symposium on Document Analysis and Information Retrieval - SDAIR95, Las Vegas, Nevada.*

Hirayama, Y. 1995. A method for table structure analysis using dp matching. In *Proc. of International Conference on Document Analysis and Recognition - ICDAR 95, Montreal, Canada.*

Hurst, M. 1999. Layout and language: Beyond simple text for information interaction - modelling the table. In *Proc. of 2nd International Conference on Multimodal Interfaces, Hong Kong.*

Itonori, K. 1993. Table structure recognition based on textblock arrangement and ruled line position. In *Proc. of International Conference on Document Analysis and Recognition - ICDAR 93.*

Kieninger, T., and Dengel, A. 1998a. A paper-to-html table converting system. In *Proc. of Document Analysis Systems - DAS 98, Nagano, Japan.*

Kieninger, T., and Dengel, A. 1998b. Table recognition and labeling using intrinsic layout features. In *Proc. of the first International Conference on Advances in Pattern Recognition - ICAPR 98, Plymouth, UK.*

Kieninger, T. 1998. Table structure recognition based on robust block segmentation. In *Proc. of the fifth SPIE Conference on Document Recognition, San Jose, California.*

Levenshtein, V. I. 1965. Binary codes capable of correcting spurious insertions and deletions of ones (original in Russian). *Russian Problemy Peredachi Informatsii* 1:12–25.

O'Gorman, L. 1992. The document spectrum for bottom-up page layout analysis. In Bunke, H., ed., *Advances in Structural and Syntactic Pattern Recognition.* World Scientific. 270 – 279.

Rahgozar, M. A.; Fan, Z.; and Rainero, E. V. 1994. Tabular document recognition. In *Proc. of the SPIE Conference on Document Recognition.*

Rice, S.; Jenkins, F.; and Nartker, T. 1995. The fourth annual test of ocr accuracy. Technical report, Information Science Research Institute (ISRI), Univ. of Nevada, Las Vegas.

Rice, S. V.; Jenkins, F. R.; and Nartker, T. A. 1996. The fifth annual test of ocr accuracy. Technical report, Information Science Research Institute (ISRI), Univ. of Nevada, Las Vegas.

Rus, D., and Summers, K. 1994. Using white space for automated document structuring. Technical Report TR 94 - 1452, Department of Computer Science, Cornell University.

Tupaj, S.; Shi, Z.; and Chang, D. C. H. 1996. Extracting tabular information from text files. Available at http://www.ee.tufts.edu/~hchang/paper1.ps.