FLORE
Repository istituzionale dell'Università degli Studi di Firenze

Rectangle labelling for an invoice understanding system

(Article begins on next page)

24 April 2024

Rectangle Labelling for an Invoice Understanding System

F. Cesarini ⋄   E. Francesconi ⋄   M. Gori *   S. Marinai ⋄   J.Q. Sheng ⋄   G. Soda ⋄

⋄ DSI, Università di Firenze, Italy, 50139; * DII, Università di Siena, Italy, 53100.
e-mail: {enrico, giovanni}@mcculloch.ing.unifi.it   http://mcculloch.ing.unifi.it/~docproc

Abstract

In this paper we present a method for the logical labelling of physical rectangles, extracted from invoices, based on a Conceptual Model which describes, as generally as possible, the invoice universe. This general knowledge is used in the semi-automatic construction of a model for each class of invoices. Once the model is constructed, it can be applied to understand an invoice instance, whose class is univocally identified by its logo. This approach is used to design a flexible system which is able to learn, from a nucleus of general knowledge, a monotonic set of specific knowledge for each class of invoices (Document Models), in terms of physical coordinates for each rectangle and related semantic label.

Keywords: *Conceptual Model; Document Model; Invoice understanding; Model Driven labelling.*

1 Introduction

This work deals with the problem of invoice understanding. We have considered the universe of invoices for the importance of this kind of documents in daily life, especially in commercial field.

The design of systems for particular reading tasks on single classes of documents has reached a consolidated level of maturity [1], [2], [3]. A more limited number of works deals with the problem of system flexibility [4], [5]. Both approaches towards document understanding have their limitations. Systems oriented to understand fixed physical or logical layouts use models that contain physical field coordinates [1] or logical relationships among fields [3]; they usually exhibit good performances in field locating, but they lack flexibility. On the other hand, systems oriented to read multi-classes of document images show a considerable flexibility, but a certain weakness in the on-line understanding phase [4], [5], [6]. These systems propose a knowledge-base oriented approach to document interpretation, which is used directly in the reading task without a predefined goal of reading. Such an approach provides these systems with a considerable degree of flexibility, however it produces a probabilistic semantic labelling of physical objects, with loss of reliability.

Some systems have been proposed in order to come to a compromise between the flexibility of open system architectures and the considerable performances of systems oriented to the fixed layout understanding. These systems express a middle policy of development among the foregoing ones. Such systems aim at constructing, for each class of documents, a model which contains object coordinates and their semantics. Moreover they provide a procedure of class recognition [7], [8].

Our work is placed in this middle policy of development, and it is an attempt to realize an invoice understanding system, which is more flexible than those oriented to the comprehension of a single type of layout, limiting the loss of performances that the enlargement of the goal can determine.

Invoices present a large variety of layouts, whose classes are usually identified by their logo. Most of information is contained in rectangular regions, limited by segments. We have considered invoices without segments inside a rectangular region and with a one-to-one mapping between logo and layout. Moreover we focused our attention on the invoice middle part, which contains information about products. The system is concerned with constructing a model, for an unknown class of invoices, that contains physical coordinates and a semantic label for each physical field containing information. This model is then used in the on-line understanding phase. The construction of a new document model is based on a Conceptual Model that represents a general knowledge about the invoice universe. The nucleus of the document model construction is represented by the individuation of rectangular regions and their labelling; labelling procedure is driven by Conceptual Model.

The invoice domain and the overall structure of a system, using the proposed labelling method, are illustrated in Section 2. In Section 3 the Conceptual

Model developed for the description of invoice world is illustrated. Section 4 describes a procedure for rectangle extraction, which is preliminary to the Document Model construction for each class of invoices. In Section 5 the semi-automatic procedure for the labelling of physical rectangles is presented. Finally in Section 6 some conclusions are reported.

2 The invoice domain

Invoices represent a universe of documents divided into classes. Even if invoices have not a fixed layout, the instances of a class are characterized by the invariance of logical and physical structure. The invoices belonging to a certain class consists of parts which we call "objects". Each of them has a logical label and contains a particular type of data. In most cases objects are surrounded by segments to form open or closed rectangles. The objects we model in an invoice are: a *Logo* which identifies the class of an invoice; *Upper*, *Middle* and *Lower Section* which contain, respectively, data related to clients, products and totals; some *Items* that are rectangular regions which contain information and sometimes the corresponding keywords.

Classes of invoices present some structural or logical similarities, which can be captured by a general knowledge, that describes logical relationships among document's objects and their physical constraints.

Nevertheless, these similarities, by themselves, are not sufficient to guarantee a robust model for understanding all the invoices. In fact, invoices have not the property of logical relationships invariance among all their objects.

In order to design a robust system to understand each invoice class and to guarantee a high degree of flexibility, we present a two level knowledge approach to invoice understanding. It is based on specific invoice class models for the on-line understanding phase and a general knowledge, which describes the invoice universe, used for a semi-automatic construction of specific models.

2.1 Invoice modelling

As we have discussed, a class of invoices is characterized by a fixed logical and physical structure. This structure can be represented by a specific model that we call *Document Model*.

A *Document Model* of a document D, consisting of objects $\{d_0, d_1, ..., d_{n-1}\}$, is the set of physical coordinates which locates univocally objects d_i, and the set of semantic labels $\{l_0, l_1, ..., l_{n-1}\}$ related to such objects.

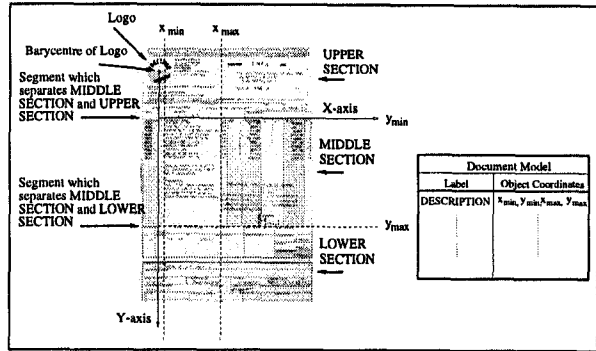Such a model may be *absolute* or *relative*. In the former case, each object has absolute coordinates with

Figure 1: **Example of invoice and related Document Model. The object** *DESCRIPTION* **and related coordinates are reported.**

respect to the scanner coordinate system. In the latter, each object has relative coordinates with respect to an object d_k of D considered as reference. We have chosen a Document Model with relative coordinates because it allows the system to be tolerant to skew variation in the on-line understanding phase. An example of Document Model is reported in Fig. 1.

A Document Model for each class of invoices is constructed by means of a higher level knowledge; we define it as *Conceptual Model*. A *Conceptual Model* for our domain is a declarative knowledge base, described by frames. Such a knowledge describes objects, or classes of objects, in terms of cardinality, observed frequency, logical and physical constraints (Section 3). Moreover, Conceptual Model represents a *part-of* hierarchy among objects. This kind of knowledge ensures flexibility to the system in the invoice domain, in the spirit of capturing some features common to the various classes, and it is used as a base for a semi-automatic procedure of Document Model construction for each class of invoices. Each Document Model constructed is mapped to the logo of the class and is used in the on-line understanding phase.

If logo is recognized, the system can locate each Item with its coordinates, under Document Model supervision. Each coordinate is related to a particular segment of those which are present in each invoice and which divide it into three logical Sections (*UPPER, MIDDLE* and *LOWER SECTION*), as it will be specified in Section 3.

If logo is not recognized, the system cannot map the invoice instance to its model. In such a case, the system deals with extracting all segments and the barycentre of logo [9]. With respect to such elements the system performs a deskewing procedure. Then segments are organized in two data structures: *Inter-*

325

val List for horizontal segments and *Interval Tree* [10] for vertical segments (Section 4.1). Hence, rectangle coordinates are extracted (Section 4.2) and the above mentioned three Sections are located (Section 4.3). Finally, a semi-automatic procedure to label each rectangle is executed (Section 5).

The result is a Document Model for the class of invoices which the current instance belongs to. Such a model can now be used for instance understanding without any other procedure, because it has been constructed on the current instance in the current position. This paper deals particularly with Document Model construction.

3 A *Conceptual Model* for invoice domain

The Conceptual Model is the result of the analysis of about thirty different classes of invoices. The Conceptual Model describes invoice domain with a declarative knowledge, based on a collection of frames. Such frames are here illustrated in an informal syntax for the sake of simplicity and immediate comprehension. Our frame scheme is inspired by the model used in [4].

Frames of Conceptual Model are arranged in a tree structure which represents the *part-of* relationship among invoice objects.

3.1 Invoice domain description

In the frames in Fig. 2 when *Observed frequency* for *Physical* or *Contextual Constraints* is not declared, it means that such constraints have been observed in the totality of instances. The object *INVOICE* is the root of the tree structure. Each *INVOICE* is divided into three sections: *MIDDLE SECTION*, where products and their codes, prizes and related taxes are described; *UPPER SECTION*, where data related to clients, their codes and conveyance modalities are reported; *LOWER SECTION*, where totals are summarized. Each of these sections is in a *part-of* relationship with *INVOICE* object.

Furthermore, each Section is composed of Items, which are in a *part-of* relationship with their Section. In this paper we concentrate on *MIDDLE SECTION* modelling, since it exhibits a certain structure regularity. As far as *UPPER* and *LOWER SECTION* are concerned, we are studying an approach based on both Conceptual Model and keyword understanding.

Examining the physical structure of *MIDDLE SECTION*, it can be observed that width is a clear discriminant feature for some Items, therefore Items of *MIDDLE SECTION* can be clustered in three Categories with respect to their width. In particular, we have decided that 1st Category cluster is to contain only the widest Item. This is a common feature of

all invoices, for which the Item *DESCRIPTION* is always the widest one in the *MIDDLE SECTION*. This feature locates univocally such an Item and it is used during the labelling phase. In general, such a clustering makes *MIDDLE SECTION* Item labelling phase easier, because it leads to a preliminary selection of the possible rectangles which may represent the Item the system is searching for (see Section 5). Examples of a frame description of some *MIDDLE SECTION* Items are in Fig. 2.

Name: DESCRIPTION	Name: QUANTITY
Data: Alphanumeric	Data: Numeric
Member of: MIDDLE SECTION	Member of: MIDDLE SECTION
Cardinality: 1	Cardinality: 1
Observed Frequency: 100%	Observed Frequency: 100%
Physical Constraints:	Physical Constraints:
-MIDDLE SECTION height;	-MIDDLE SECTION height;
-1st Category width.	-2nd Category width (70%);
Contextual Constraints:	-3rd Category width (30%).
-None.	Contextual Constrains:
	-If QUANTITY $\in 2^{nd}$ Category $\Rightarrow$ QUANTITY on the right of DESCRIPTION;
	-If QUANTITY $\in 3^{rd}$ Category $\Rightarrow$ QUANTITY on the left of DESCRIPTION;
	-On the left of TOTAL.

Figure 2: **Examples of a frame description of two** *MIDDLE SECTION* **Items**

It can be observed that the Item *QUANTITY*, in different classes of invoices, may satisfy two different Category physical constraints. Depending on the Category it belongs to, it has different contextual relationships with other Items. The tree structure for *MIDDLE SECTION* is shown in Fig. 3.
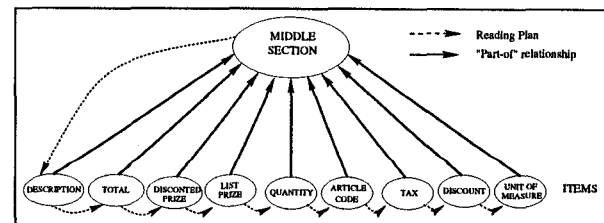

Figure 3: **Tree structure for** *MIDDLE SECTION*.

In the tree structure, Items of *MIDDLE SECTION* are ordered with respect to their contextual dependencies, to a decreasing observed frequency and to the priority of reading. Such an order gives a contribute for locating objects in the labelling phase of

MIDDLE SECTION. Providing a priority of reading without back-tracking, it is able to greatly reduce the search space at each step, enhancing the likelihood of right location of following Items. We refer such an order as *Reading Plan.*

4 Extraction of physical features

As discussed in Section 2, if logo is not recognized, the system provides a procedure of semi-automatic Document Model construction. The first step of such a procedure deals with extracting the invoice physical features (segments and rectangles).

4.1 Segment extraction and organization

A common feature of invoice domain is represented by the presence of horizontal and vertical segments to form rectangles which limit each Item. In particular, the system is concerned with invoices without segments inside an Item and with segments that surround, entirely or partially, each Item, to form physical or virtual rectangles.

Horizontal and vertical segments are extracted with techniques of low level processing [11]. Ordinates of horizontal segments are organized, without repetitions, in a linked list whose elements contain such ordinates in an increasing order. Each element is associated with another linked list, whose elements contain starting and final abscissas of each horizontal segment related to the considered ordinate. The succession of two consecutive elements in the first level linked list represents the interval between the two sets of horizontal segments related to the ordinates of such elements. For this reason we define this structure as *Interval List*. The area bounded by the two sets of horizontal segments related to two consecutive ordinates is called *vertical interval* (Fig. 4).

Vertical segments are organized in a binary tree. Such a tree is called *Interval Tree* [10]; its leaves contain, without repetitions, abscissas of vertical segments. The succession of two consecutive nodes at the same level represents an interval (*horizontal interval*) between the two sets of vertical segments related to the abscissas of such nodes. In our work Interval Tree structure has been modified in order to consider all vertical segments which limit each horizontal interval, and to permit a linear parsing of leaves. Each leaf is linked to the next one and is associated with a linked list, whose elements contain starting and final ordinates of each vertical segment related to the current abscissa. An example of the structures we use is presented in Fig. 4.

4.2 Algorithm for rectangle extraction

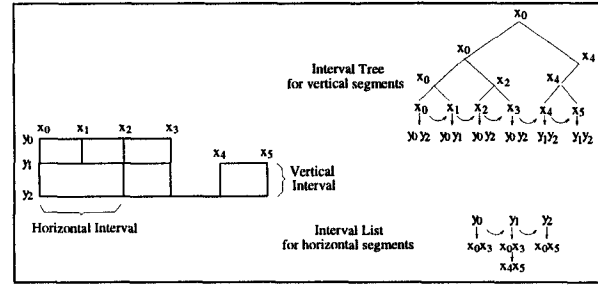In this section we will describe the algorithm for rectangle extraction, Fig. 5 can be used for reference.

Figure 4: **Segment collections structured in** *Interval List* **and in** *Interval Tree.*

A physical rectangle can be identified by four coordinates $(x_{min}, x_{max}, y_{min}, y_{max})$. From invoice domain point of view, a rectangle is a physical area limited, at least, by two portions of horizontal segments. The two portions are to belong to different horizontal segments with different ordinates. The two portions are to have the same starting and final x-coordinate, respectively x_{p_start} and x_{p_stop}, and their lengths have to exceed a threshold. Abscissas of such portions are obtained by the intersection, on X-axis, between the projections of the current horizontal segment, with ordinate y_i, and the following ones with ordinate y_j, $y_j > y_i$. We denote by $H_{i,j}$ the jth horizontal segment of the ith ordinate.

The algorithm which performs rectangle extraction is based on the individuation of such an intersection, that determines the coordinates x_{p_start} and x_{p_stop} of a partition of the current horizontal segment $(H_{i,j})$. In this rectangular area, identified by the coordinates $(x_{p\_start}, x_{p\_stop}, y_i, y_j)$, eventual further subdivisions into rectangles are individuated by the presence of vertical segments which completely cover the current vertical interval (y_i, y_j). The abscissa x_k of such a vertical segment determines an x-coordinate of a rectangle.

The portion of the current horizontal segment individuated by the interval $(x_{p\_start}, x_{p\_stop})$ is allocated to the rectangles that it locates, and it is not considered any more.

In each vertical interval other eventual vertical segments, internal to a rectangle, which do not completely cover the current vertical interval are located.

All rectangles extracted are organized in a *Four Directional Adjacency Graph* (FDAG) [12], because this structure guarantees more nimbleness than linear or hierarchical spatial data structures in dealing with neighborhood problems. Each rectangle is represented by a node of FDAG which contains x and y coordinates $(x_{min}, x_{max}, y_{min}, y_{max})$. The node of FDAG corresponding to a rectangle is associated with
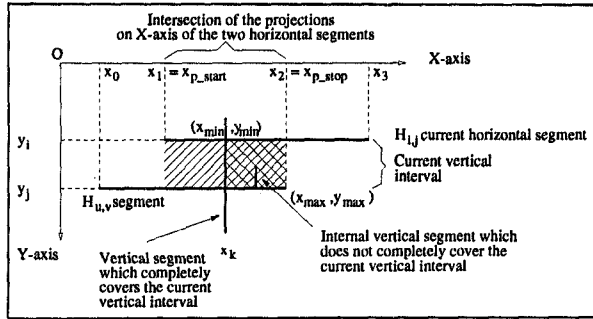
Figure 5: **Intersection on X-axis of the projections of two horizontal segments**

a linked list whose elements contain the coordinates of physical segments which limit such a rectangle and, in case, of the vertical internal segments. The algorithm for rectangle extraction is summarized in the following:

Algorithm for rectangle extraction

Input: Segments organized in Interval List and Interval Tree;

Output: Rectangles organized in a Four Directional Adjacency Graph (FDAG);

Begin
- Until all elements y_i of the *Interval List* are visited;
  Begin
  - Until all horizontal segments $H_{i,j}$ with y_i ordinate are visited;
    Begin
    - Until the current horizontal segment $H_{i,j}$ is completely allocated to a rectangle or all the segments $H_{u,v}, u > i, v \geq 0$ are visited;
      Begin
      - The intersection of the projections on X-axis of $H_{u,v}$ and the portion of $H_{i,j}$ not yet allocated to a rectangle is calculated; such an intersection determines a partition of the current vertical interval; y_i and y_j are y_{min} and y_{max} of all rectangles in the current vertical interval;
        - In the current partition of vertical interval, all vertical segments are extracted (abscissas of vertical segments which completely cover the current vertical interval determine the x_{min} and x_{max} boundary abscissas of rectangles);
        - For each extracted rectangle, a new node for the FDAG, containing rectangle coordinates, is constructed; the internal and boundary segments of the rectangle are inserted in the list associated to the FDAG node.
        - FDAG is updated by the new nodes;
      End
    End
  End
End

4.3 Location of the Sections

The three Sections every invoice consists of, according to the Conceptual Model, are always individuated by two horizontal segments. Their localization only

requires the analysis of *Interval List* collecting all horizontal segments. The two horizontal segments, among the N of the collection, are selected by a heuristic criterion, which provides of searching for two consecutive horizontal segments $H_{\bar{i}}$ and $H_{\bar{i}+1}$ of equal length (minus an error (ϵ)), greater or equal than 80% of document width. Moreover the distance between $H_{\bar{i}}$ and $H_{\bar{i}+1}$ is to be the greatest distance between any other two consecutive segments, and such distance is to be greater or equal than 30% of the distance between the first and the last horizontal segment of the collection. The selected vertical interval $(y(H_{\bar{i}}), y(H_{\bar{i}+1}))$ is the *MIDDLE SECTION*.

In some classes of invoices a sequence of rectangles which contain the keywords of related rectangles of *MIDDLE SECTION* may be present. Each of these rectangles is in a vertical dependence relationship [8] with a rectangle of *MIDDLE SECTION*, and all these rectangles are to have the same height (Fig. 6). Moreover all these rectangles are to be located in the same vertical interval.

Article code	Description	Qty	Prize	Amount

Figure 6: **Vertical dependence relationships among rectangles**

In order to consider this possibility, the system verifies the presence of vertical dependences for the rectangles of *MIDDLE SECTION* with respect to the superior adjacent rectangles. If the condition is verified, rectangles of the superior sequence may be used to search for the keywords during the labelling procedure of rectangles of *MIDDLE SECTION* (see Section 5).

The coordinates of the two segments, which locate the three Sections, and their logical labels are reported in the Document Model. One of such segments is elected as reference primitive, to be used during the on-line understanding phase. Physical coordinates of rectangles, extracted in every Sections, are reported in the Document Model with respect to the segment elected as reference primitive.

5 Labelling procedure

After rectangles have been located, the system attempts to attribute them a logical label. In this paper, a Model Driven labelling strategy for *MIDDLE SECTION* is proposed.

For each Item provided by the Reading Plan, such a strategy is concerned with searching for a rectangle

which satisfies the physical and contextual constraints, indicated in the corresponding frame.

Section 5.1 illustrates the model driven labelling procedure we propose. After the end of this procedure, some rectangles might be not labelled. In this case recognition of keywords must be performed for their labelling, as illustrated in Section 5.2.

5.1 Model Driven labelling

After all invoice rectangles have been extracted and *MIDDLE SECTION* has been located, a procedure for clustering its rectangles is executed. Such a procedure is provided with a constraint related to the presence of only one rectangle in the 1st Category cluster.

According to the Reading Plan, *DESCRIPTION* is the first Item to be searched for. Its properties locate it univocally, without any reference to the positions of rectangles already labelled. Such an Item results a minimally dependent concept [5], therefore it is the first Item to be located in the labelling phase. According to *DESCRIPTION* physical constraints (see Fig. 2), the system selects the rectangles in the 1st Category cluster. Since the clustering procedure provides only one rectangle in this cluster, such a rectangle is labelled as *DESCRIPTION*.

The localization of other Items is based on Items previously located. In our case Item *TOTAL* refers to the position of *DESCRIPTION*, *QUANTITY* refers to *TOTAL* and *DESCRIPTION* positions, and so on with succeeding Items. The order of localization of the Items is suggested by the Reading Plan.

In order to locate Items, the system searches for a rectangle, related to the current Item suggested by the Reading Plan. A series of rectangles, satisfying a number of constraints of the current Item, are selected. The number of constraints satisfied provides a credibility order among rectangles selected. Rectangles selected are proposed to the user in a decreasing credibility order, till a rectangle is accepted for the current semantic label.

For Items with alternative physical constraints (e.g. Item *QUANTITY*), the system firstly searches (and possibly proposes to the user) rectangles satisfying the constraint with the highest observed frequency, and then the rectangles satisfying the other constraints.

User interaction, in confirmation or refutation of a labelling hypothesis, is justified by the fact that rectangle semantic labelling is a "key" procedure in a Document Model construction. An error in the labelling phase of a rectangle, in fact, spreads to the others, with the risk of making every other semantic attributions fail.

If none of the rectangles selected is accepted by the user for the current semantic attribute, it means that the related Item is not present in the current invoice class, or no rectangle satisfies the constraints provided by the Conceptual Model for the current Item.

5.2 Keyword based labelling

At the end of the Reading Plan, if some unlabelled rectangles are left, a labelling procedure based on reading the keywords related to each rectangle, is executed. The keywords related to the rectangles of *MIDDLE SECTION* are located in the first row of the rectangles or in the adjacent upper rectangles (Fig. 6). The reading phase is executed by an OCR system. Each OCR output is validated by a dictionary that contains the possible keywords belonging to Items of *MIDDLE SECTION*. Such a dictionary, that may be of large dimensions, is semantically partitioned with respect to the Items of *MIDDLE SECTION*. As a matter of fact, invoices belonging to different classes may associate different keywords to the same semantic information (e.g., "Quantity" or "QTY" for the Item *QUANTITY*).

This labelling phase, based on a bottom-up strategy, is partially driven by the Conceptual Model as well. In fact, when the Reading Plan is over, and there are some rectangles not labelled, we can have two possibilities:

1) all the Items provided by the Reading Plan are assigned to a rectangle. In this case the Conceptual Model doesn't provide any label for the rectangles left and it should be updated.

2) there are some Items not assigned to any rectangles. In this case the system attempts to match the keyword of the current unlabelled rectangle with the alternatives suggested by the dictionary for the not assigned Items. We can have two possibilities:

• the keyword matches with one of the dictionary alternatives related to an Item not assigned: the related semantic label is assigned to the current rectangle;

otherwise:

• it could be the case of a new keyword for an Item already present in the Conceptual Model, or the case of a keyword corresponding to a not present Item. Therefore the system shows the user all the not assigned Items: if the user chooses one of them, only the dictionary is updated by the new keyword. If none of them is assigned to a rectangle a new semantic has to be assigned, then the Conceptual Model and the dictionary have to be updated.

The scheme of the labelling procedure is in Fig. 7.

The aim of the labelling procedure is to resort to keyword reading as less as possible and, in case key-
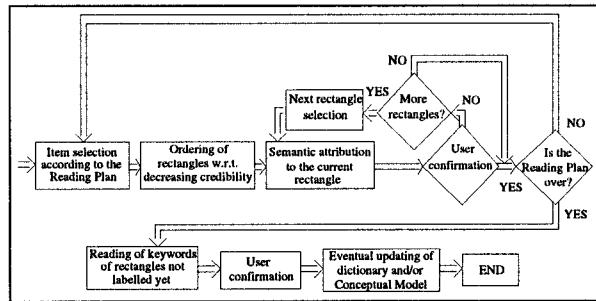
329

Figure 7: **Scheme of the Item labelling procedure**

word reading is necessary, it avoids consulting a large dimension dictionary for their validation.

6 Conclusions

In this paper a semi-automatic labelling method of physical rectangles, extracted from the Middle Section of an invoice instance of an unknown class, is presented. Such a method is based on a Conceptual Model that describes some structural or logical similarities of the invoice domain. We obtain a Document Model for the Middle Section of each class of invoices, which contains physical coordinates of the Items and their semantics, and, globally, a monotonic set of knowledge, oriented to the understanding of the Middle Section of invoices.

The extension of this kind of approach to the other sections of the invoice is under consideration.

The labelling approach we propose spares the user to indicate explicitly the rectangles containing information and keywords, and, in most cases, to write explicitly the semantic of such rectangles. As a matter of fact, the approach limits user interaction to only confirmation or refutation of a suggested semantic attribution. Moreover, in most cases, such labelling procedure spares the system to locate and read the keywords. Note that, in some cases, keywords might not be present.

The proposed labelling approach is part of an Invoice Understanding System, that we are developing at the DSI (Dipartimento di Sistemi e Informatica, University of Florence). The system is under implementation on a Sun Sparc system.

References

[1] F. Cesarini, M. Gori, S. Marinai, and G. Soda, "A system for data extraction from forms of known class," in *Proc. of the Int. Conference on Document Analysis and Recognition*, pp. 1136–1140, 1995.

[2] Y. Y. Tang, C. Y. Suen, C. D. Yan, and M. Cheriet, "Financial document processing based on staff line and description language," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 25, no. 5, pp. 738–753, 1995.

[3] T. M. Ha and H. Bunke, "Model-based analysis and understanding of check forms," *Int. Journal of Pattern Recognition and Artificial Intelligence*, vol. 8, no. 5, pp. 1053–1081, 1994.

[4] S. W. Lam, "An adaptative approach to document classification and understanding," in *Document Analysis Systems*, pp. 231–252, 1994.

[5] F. Esposito, D. Malerba, G. Semeraro, and M. Pazzani, "A machine learning approach to document understanding," in *Proc. of the 2nd Int. Workshop on Multistrategy Learning*, pp. 276–292, 1993.

[6] M. Lipshutz and S. L. Taylor, "Functional decomposition of business letters," in *Symposium on Document Analysis and Information Retrieval*, pp. 435–448, 1995.

[7] A. Dengel, R. Bleisinger, R. Hoch, F. Fein, and F. Hones, "From paper to office document standard representation," in *IEEE Computer*, pp. 63–67, 1992.

[8] T. Watanabe, Q. Luo, and Sugie, "Layout recognition of multi-kinds of table-form documents," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 17, no. 4, pp. 432–445, 1995.

[9] F. Cesarini, M. Gori, S. Marinai, and G. Soda, "A hybrid system for locating and recognizing low level graphic items," in *Graphics Recognition - Methods and Applications*, pp. 135 – 147, Springer-Verlag, LNCS 1072, 1996.

[10] H. Samet, "Hierarchical representations of collections of small rectangles," *ACM Computing Surveys*, vol. 20, no. 4, pp. 271–309, 1988.

[11] A. K. Chhabra, V. Misra, and J. Arias, "Detection of horizontal lines in noisy run length encoded images: The fast method," in *Graphics Recognition - Methods and Applications*, pp. 35–48, Springer Verlag, LNCS 1072, 1996.

[12] C. Y. S. J. Yuan, Y. Y. Tang, "Four directional adjacency graphs (fdag) and their application in locating fields in forms," in *Proc. of the Int. Conference on Document Analysis and Recognition*, pp. 752–755, 1995.