Construction of Model of Structured Documents Based on Machine Learning

Sergey Golubev[1,2]

[1] Moscow Institute of Physics and Technology, Dolgoprudny, Russia
[2] ABBYY Software, Moscow, Russia
Gergey_G@abbyy.com

Abstract. In this paper we consider the problem of structured document recognition. The document recognition system is proposed. This system incorporates a recognition module based on methods of structured image recognition, a graph document model and a method of document model generalization. The machine learning component makes the process of document model construction easier and less time-consuming.

Keywords: document recognition, machine learning, graph document model.

1 Introduction

The extensive development of information technologies and electronic document management systems poses the problem of converting paper documents into digital form. While modern OCR systems can recognize symbols with high precision, simple character recognition is often insufficient. To capture the data correctly, a recognition of logical document structure is needed. The most precise and flexible methods of recognition of document structure are those based on structured pattern recognition [1,2]. These methods use the model of document structure which is compared to the document image during recognition.

The most widespread document model is the graph model where text blocks and separator lines form graph vertices, whereas graph edges correspond to relations between them [3]. The recognition problem in this case is the problem of matching of two graphs: the model graph and the document graph which is formed from the document image prior to recognition process. Systems based on graph models often allows automatic construction of document model based on methods of machine learning.

Another document recognition system based on the methods of structured pattern recognition is the ABBYY FlexiLayout system [4]. This system uses more complex special-purpose document model (structural description). In comparison with graph based methods the FlexiLayout system can reach higher recognition quality especially for documents with complex structure. The main drawback of the ABBYY FlexiLayout system is that the document description is designed to be created manually which is difficult and time-consuming. It seems reasonable to combine the FlexiLayout document recognition system with the methods of machine learning. This allows us to take advantage of both the precision of the FlexiLayout system and the easy constructing of graph based document model. We propose a trainable system for document recognition which incorporates the FlexiLayuot recognition module and a training module. The training module uses an intermediate graph document model on the training stage. The graph model is then converted into the FlexiLayout document description. In this paper we will discuss the graph document model and the process of its construction.

2 Problem Setting

In this work a special class of documents, so called forms, is discussed. The distinctive feature of this document class is that the document structure consists of the static part (which is presented by headings, separator lines etc.) and data elements. The examples of forms are various questionaries, financial documents like invoices, payment orders, etc.

All forms can be divided in four classes in terms of structure complexity: fixed forms, semi-fixed forms, flex-forms and free forms. A fixed form is a document that maintains all characteristics of its layout (except for scanning distortions such as stretching, skew etc.). The recognition of fixed-forms is a relatively simple task and does not require structure recognition. Semi-fixed forms are created from fixed-forms by relaxing some restrictions on the document layout. An example of semi-fixed form is an electronic template which allows to move succeeding data fields, if more space is needed for the given field or section. The recognition of semi-fixed forms can be performed using table or graph based document models. In a flex-form (also called context-form) each data element has an identifiable local context associated with it. The data element and the related context may be located in varying places on the document image. However, the relative position of the data element to its context is relatively constant. Free forms include documents which are not fixed, semi-fixed or flex-forms, extraction of data from this documents may require advanced natural language processing. In current paper we will consider the first three types of forms: fixed, semi-fixed and flex-forms.

The problem of form recognition is the following: given a document image the system must detect the localization of data elements so as to recognize the required data (using OCR technologies) and input it into the database. The system uses certain number of document images with marked locations of data elements as training set. After the training process the system must be able to locate the data elements on any other document image of the same document type.

The recognition problem is solved in two stages. On the first stage a model is generated on the base of training examples, on the second stage this model is used to solve recognition (prediction, classification) problem. In our work we use the ABBYY FlexiLayout system as a recognition module. Thereby the model is the FlexiLayout structural description (for more information on structural descriptions see [4]). The FlexiLayout document model contains the descriptions of the document structural elements. The description of each structural element defines its attributes, allowing to detect the structural element on the document image. Unlike the graph based methods the FlexiLayout system does not use *a priori* extraction of structural elements. Instead, structural elements are detected directly during the recognition process according to the document description and taking into account the particular features of a document type. Particularly important feature of the system is the ability to calculate attributes of structural elements and relations between them "on the fly". For example the distance between two elements may be calculated using the size of detected rectangle of one of the elements. The relative positions of structural elements are described in the model in terms of metrical and ordering relations. In this respect the FlexiLayout model is similar to graph models. In fact a graph model with metrical relations may be converted into the FlexiLayout description if the labels of graph vertices have the appropriate format. The FlexiLayuot model also supports fuzzy relations between elements.

The document description uses a specialized language to describe attributes of structural elements and relations between them. Therefore it is not convenient for machine learning and it was designed mainly to be created manually by system operator. Therefore, it is preferable to use a simpler graph model on the stage of model generation. When the intermediate graph model is generated in the learning process it is then converted into FlexiLayout structural description. FlexiLayout description in then used for document recognition. In the following section we will discuss the intermediate graph document model and the method of its generalization.

3 Document Model

In structural pattern recognition methods it is convenient to represent the document image as a set of image objects. The input document image is a raster image obtained by scanning the paper document. This image is subjected to segmentation process which extracts connected pixel components. Connected components are the images of separate letters, punctuation symbols, parts of separator lines and pictures. Each connected component is classified and recognized (if it represents letter). We use classified and recognized connected components as elementary image objects. The elementary image object is described by its type, surrounding rectangle and Unicode code of recognized symbol (for letters, diacritics and punctuation symbols). The image object is either an elementary object or a combination of elementary objects.The complex image object is described by its constituents. The following hierarchy of image objects is used:

1. Letter $\rightarrow$ Word $\rightarrow$ Text line $\rightarrow$ Text fragment
2. Separator fragment $\rightarrow$ Separator line
3. Picture fragment $\rightarrow$ Picture

The intermediate representation of the document structure uses a graph model, which allows us to take advantage of well-known methods of graph data processing [5,6,8]. Document model in our case should be able to represent both single document image and generalized document structure. In the first case it is called a *document graph* and in the second case, a *template graph.*

The vertices of the graph correspond to image objects in the case of document graph or to elements of logical structure in the case of generalized description. Graph edges correspond to metric relations between objects. Thereby the document model is an oriented labeled graph with labels on vertices and edges: $G_{\mathrm{D}} = (V, E, L_{\mathrm{V}}, L_{\mathrm{E}}, f_{\mathrm{V}}, f_{\mathrm{E}})$, where V is the set of vertices, E is the set of edges, L_{V}, L_{E} are sets of vertex and edge labels respectively, f_{V}, f_{E} are mappings taking vertices and edges to their labels.

Vertex labels have the following form: $l_{\mathrm{V}} = (id, description)$, where:

id is an image object or structural element identifier, describing its logical role in document structure, which can possess values from the set $R, S, F_1, F_2, , F_N$, where:

F_i are identifiers of data elements. Since data elements set is known for particular document type and positions of data elements are marked on images from training set, the objects which correspond to data elements have unique identifiers.

R is an identifier of static text object.

S is an identifier of separator line .

description is a description of the set of image objects, which can represent the element of logical structure on the image. This set can be defined in several ways: the set consisting of single image object, the set of phrases consisting of given keywords, the set of text strings consisting of characters from a given alphabet, etc.

Edge labels have the following form: $l_{\mathrm{E}} = (xDistamce, yDistance)$, where $xDistance$, $yDistance$ — distance between rectangles measured horizontally and vertically, respectively and represented as intervals. This distance may be positive or negative depending on vertex order so the edges of document graph are directional.

4 Document Model Generalization

Consider the following definition. A template graph G_{T} *describes* a document graph G_{D} if there exists a mapping $X : V(G_{\mathrm{T}}) \to V'(G_{\mathrm{D}})$, where $V'(G_{\mathrm{D}}) \subseteq V(G_{\mathrm{D}})$, and for vertices $v, u \in V(G_{\mathrm{T}})$ and their images $v', u' \in V'(G_{\mathrm{D}})$ the following conditions are fulfilled:

1. $id(v) = id(v')$, i.e. vertices have the same identifiers.
2. Object described by the label of vertex v', is a member of the objects set described by the label of vertex v.

3. The label l of vertex $e = (v, u)$ is the generalization of the label l' of vertex $e' = (v', u')$, in the sense that *distance* intervals of the label l are comprised in *distance* intervals of the label l'.

We will perform construction of template graph by generalizing it successively with document graphs from the training image set. Let G_{T} be a template graph constructed on some document set and G_{D} be a document graph of the next document in the training set. A template graph can be generalized from the graph G_{D} in the following way. One need to build the mapping of template graph vertex subset to document graph vertex subset: $Y : V'(G_{\mathrm{T}}) \to V'(G_{\mathrm{D}})$, where $V'(G_{\mathrm{T}}) \subseteq V(G_{\mathrm{T}})$, $V'(G_{\mathrm{D}}) \subseteq V(G_{\mathrm{D}})$, so that identifiers of image vertices and counter image vertices match. After that the labels of vertices from set $V'(G_{\mathrm{T}})$ should be generalized with the labels of vertices from set $V'(G_{\mathrm{D}})$ so that object sets described by vertex labels comprise new objects described by vertex labels from $V'(G_{\mathrm{D}})$. Similarly the intervals of edge labels of graph G_{T} should be generalized with the intervals of corresponding edge labels of graph G_{D}.

Obviously there is a large number of such mappings and corresponding template graphs. We must define some measure to choose the most appropriate template graph. It should be noted that discussed vertex mapping corresponds to an edit path between graphs [5]. In general a probabilistic measure can be used for edit path [7], however in our case it is reasonable to use specific measure which is more relevant to our problem setting.

Assume the following measure for vertex mapping and the corresponding template graph G'_{T}:

$$Q = \left(\prod Q_{\mathrm{V}}(v_i \to u_j)\right) * Q_{\mathrm{U}}(V(G_{\mathrm{T}}) \setminus V'(G_{\mathrm{T}})) * Q_{\mathrm{U}}(V(G_{\mathrm{D}}) \setminus V'(G_{\mathrm{D}})), \quad (1)$$

where Q_{V} is the measure of pair of vertices from mapping Y, Q_{U} is the measure of sets $V(G_{\mathrm{T}}) \setminus V'(G_{\mathrm{T}})$ and $V(G_{\mathrm{D}}) \setminus V'(G_{\mathrm{D}})$, i.e. sets of vertices without image or counter image. The measure Q_{U} depends on the number of elements in the corresponding set and has the following form: $Q_{\mathrm{U}}(V(G) \setminus V'(G)) = U^{|V(G) \setminus V'(G)|}$, i.e. for every missed vertex a constant penalty U is given. Since vertex identifiers are the same for image and counter image vertices, correspondence between document data elements is unambiguous and we should consider only static objects in template graph measure.

$$Q_{\mathrm{V}}(v \to u) = Q_{\mathrm{V}}(v') = (1 - \alpha) * \max Q_{\mathrm{E}}(e_i) + \alpha * \sum Q_{\mathrm{E}}(e_i)/N_{\mathrm{F}}), \quad (2)$$

where $Q_{\mathrm{E}}(e_i)$ is the measure if i-th edge, coming from vertex v' of graph G'_{T}, which is the result of matching vertex v of graph G_{T} to vertex u of graph G_{D}, $i \in [1, N_{\mathrm{F}}]$; N_{F} is data elements count in current document type; α is an empirical factor.

Edge measure has the following form:

$$Q_{\mathrm{E}}(e_i) = \max\left(0; 1 - \left(\frac{\Delta x + \Delta y + 4 - \sqrt{(\Delta x - \Delta y)^2 + 16}}{2}\right)^2\right), \quad (3)$$

where $\Delta x = Width(xDistance)/W$, $Width(xDistance)$ is the width of the interval $xDistance$ if edge e_i, W is the characteristic interval width which is about page size; the same is for Δy. This empirical measure is chosen from the following considerations. First, the function isolines must be concave, i.e. the measure of edge with intervals (for example) $\Delta x = 1$, $\Delta y = 0$ should be higher than measure of edge with intervals $\Delta x = 1/2$, $\Delta y = 1/2$. Second, the measure must be close to 1 for edges resulting from correct object matching (Δx and Δy are sufficiently less than 1), and decline substantially while Δx and Δy get closer to 1. In our case the isolines are hyperbolas $y = \frac{4}{x+a} - a$, $a \in [1, 2]$. The edge measure function plot in area $\Delta x \in [0, 1]$, $\Delta y \in [0, 1]$ is a surface obtained by "sliding" hyperbola $y = \frac{4}{x+1} - 1$, $z = 0$ along parabola $z = 1 - x^2$, $x = y$.

With template measure being defined, the problem is to find a vertex mapping between graphs with the best measure. This can be done in the following way:

– We will sequentially choose corresponding pairs for vertices of template graph using search tree. Search tree node corresponds to decision of matching vertex v_i to vertex u_j or leaving vertex v_i without match.
– For each nonterminal node and the corresponding tree path we will define partial measure Q_{P}, which in contrast to Q does not take into account unmatched vertices, i.e. $Q_{\mathrm{U}}(V(G_{\mathrm{T}}) \setminus V'(G_{\mathrm{T}}))$ depends only on examined vertices of template graph, and $Q_{\mathrm{U}}(V(G_{\mathrm{D}}) \setminus V'(G_{\mathrm{D}}))$ is omitted.
– The tree search is performed in the order of descending of path measure. Since the path measure cannot increase with adding new nodes to the path, then the measure of current path is an upper estimate of all extensions if this path. Thus, the search method may be based on the Dijkstra's algorithm for trees.

5 Experimental Evaluation

Efficiency of suggested recognition method was tested using cross-validation technique with increasing size of the training set. Testing document set was sorted randomly. On each iteration the first n documents were used as the training set and the $(n + 1)$-th document was used to test recognition quality. The percentage of recognition errors was calculated. The test was performed several times with different sorting of document set, then the average error percentage for n-th iteration was calculated. This allows us to estimate the required size of the training set and the minimum error percentage the system can reach. Fig. 1 shows the dependence of error percentage on the size of the training set.

It can be seen that for semi-fixed forms with moderate structure variations the system needs 5-10 training samples to achieve sufficient precision of about 95% (5% of errors). For more complex forms with grater structure variations (so called flex-forms) larger training set is needed (at least 20 document samples). The recognition precision in this case is only 75-80%. This in insufficient for using the system in the automatic mode. However, since the result of training process is the FlexiLayout document description, the document model may still be adjusted manually. This allows one to increase recognition quality to appropriate level.

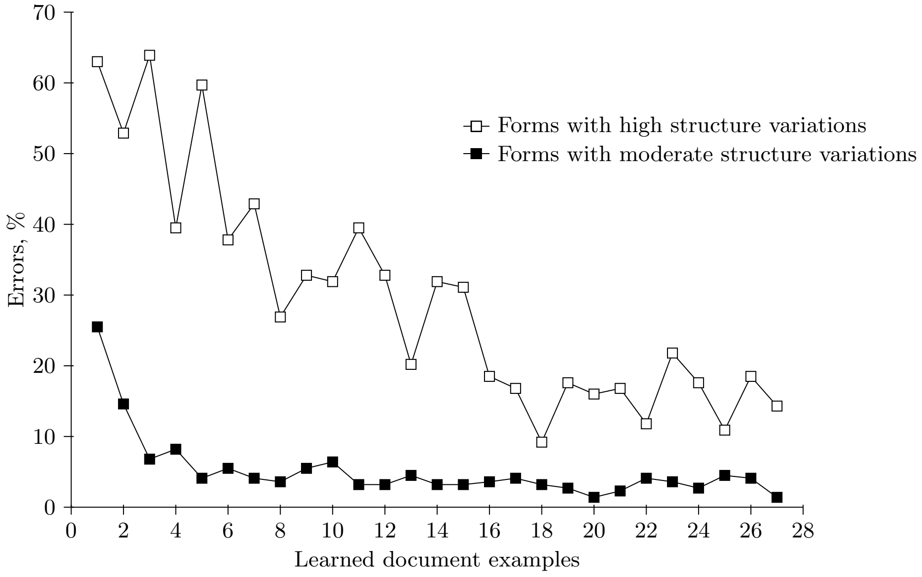

Fig. 1. Dependence of recognition errors percentage on training set size

6 Conclusion

A graph document model and structured documents recognition framework was proposed. We showed that the application of machine learning techniques in FlexiLayout document recognition system allows one to significantly reduce time costs for creating document model while maintaining sufficient recognition quality. Experiments showed that automatically generated document model allows the FlexiLayout system to recognize semi-structured documents with high precision. In case of flex-forms with significant structure variations the document description must be adjusted manually after being generated. However, even in this case the usage of automatic construction of document model results in considerable time saving. The further study will be aimed at increasing the recognition quality for complex documents with high structure variations.

References

1. Farrow, G.S.D., et al.: Model Matching in Intelligent Document Understanding. In: Proc. of ICDAR 1995 (1995)
2. Hirayama, Y.: Analyzing Form Images by Using Line-Shared-Adjacent Cell Relations. In: Proc. of IAPR 1996 (1996)
3. Yuan, J., Tang, Y.Y., Suen, C.Y.: Four Directional Adjacency Graphs (FDAG) and Their Application in Locating Fields in Forms. In: Proc. of ICDAR 1995 (1995)
4. Zuyev, K.A.: System for Identification of Structure of Printed Documents, Candidate of Science Dissertation, MGUL (in Russian) (1999)

5. Cook, D., Holder, L.: Mining Graph Data. Wiley Interscience, Hoboken (2006)
6. Kuramochi, M., Karypis, G.: An efficient algorithm for discovering frequent subgraphs, Tech. Rep. 02-026 Minneapolis, University of Minnesota (2002)
7. Neuhaus, M., Bunke, H.: A probabilistic approach to learning costs for graph edit distance. In: Proceedings 17th International Conference on Pattern Recognition, vol. 3 (2004)
8. Yan, X., Han, J.: gSpan: Graph-Based Substructure Pattern Mining. In: Proc. IEEE International Conference on Data Mining (ICDM 2002), Los Alamitos (2002)