Configuration REcognition Model for Complex Reverse Engineering Methods: *2(CREM)*

Karim Hadjar, Oliver Hitz, Lyse Robadey, and Rolf Ingold

DIUF, University of Fribourg,
Chemin du Musée 3, 1700 Fribourg, Switzerland,
{firstname.lastname}@unifr.ch

Abstract. This paper describes *2(CREM)*, a recognition method to be applied on documents with complex structures allowing incremental learning in an interactive environment. The classification is driven by a model, which contains a static as well as a dynamic part and evolves by use. The first prototype of *2(CREM)* has been tested on four different phases of newspaper image analysis: line segment recognition, frame recognition, line merging into blocks, and logical labeling. Some promising experimental results are reported.

1 Introduction

In the field of document recognition many improvements have been made during the last decade. However, there is still a lack especially in recognizing complex structured documents, such as newspapers or magazines.

The first layout analysis methods were focused on simple document structures [1]. Some recent works show a great interest in complex layout analysis [2, 3,4], and introduce the concept of learning-based algorithms [5].

Currently known approaches rely on document models, which are either set up by hand or generated automatically in a previous learning step that needs a lot of ground-truthed data [6]. The drawback is that such models do not accommodate easily to new situations, a condition that is very important when dealing with complex document structures.

Human beings are able to cope with new situations using their knowledge. They learn from experiences and improve their knowledge incrementally. Given the variety of different types of documents, the same strategy should be applied to document recognition by computers.

We believe that *interactive incremental learning* is an important issue for document recognition. It is one of the main goals of the CIDRE[1] project, which aims at building a semi automatic document recognition system that constructs its knowledge incrementally through the interaction with the user. Our previous work was devoted to software architecture issues [7,8] as well as logical structure and font recognition algorithms [9,10].

[1] CIDRE stands for Cooperative and Interactive Document Reverse Engineering and is supported by the Swiss National Fund for Scientific Research, code 2000-059356.99-1.

D. Lopresti, J. Hu, and R. Kashi (Eds.): DAS 2002, LNCS 2423, pp. 469–479, 2002.

In this paper we introduce *2(CREM)*, which stands for Configuration REcogntion Model for Complex Reverse Engineering Methods. It is an interactive recognition method for complex structured documents based on incremental learning.

This paper is organized as follows: in section 2 we introduce the principles of *2(CREM)*. Section 3 is devoted to the experimental part and the results obtained. Finally in section 4 we conclude our work and give some perspectives for future work.

2 *2(CREM)* Principles

The goal of *2(CREM)* is to classify document primitives named "objects". Depending on the application, these objects can be line segments, frames, text lines, etc.

An object is associated with different attributes such as position and size, font definition, neighborhood relation etc. Not all attributes are necessary to classify an object. Some classification tasks use a subset of the available attributes. In *2(CREM)*, such attributes are called "relevant attributes". They are defined in the static part of a *2(CREM)* model, as illustrated in figure 1. The static part therefore defines all possible features that can be used for classification.

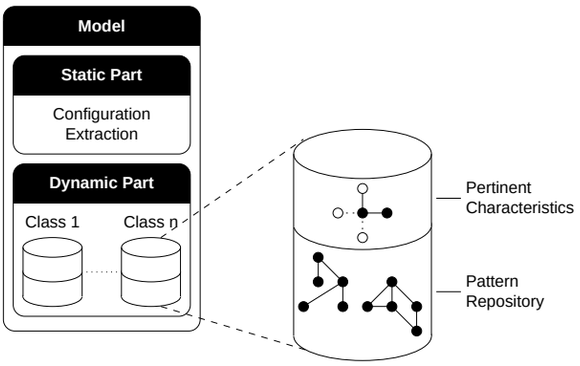

Fig. 1. The model of *2(CREM)*.

The set of features is called "configurations". In the dynamic part of a model, reference configurations are stored for every class. These configurations represent the knowledge of the model and are called "patterns".

A configuration consists of different kinds of characteristics associated to the object to be classified or to other objects referred to by neighborhood relations. The dynamic part contains a set of relevant attributes per class.

The classification procedure of the *2(CREM)* is illustrated on figure 2. For every object to classify, we extract a configuration according to the static part

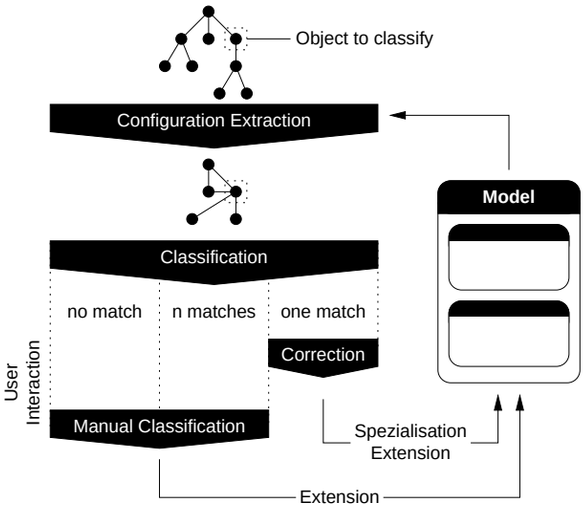

Fig. 2. Architecture of *2(CREM)*.

of the model. This configuration is then compared to the patterns of each class, by taking into account the pertinent characteristics.

This comparison produces three possible alternatives:

- The configuration matches exactly one class: this is the ideal case. The system classified the object. User interaction is possible to correct erroneous classifications.
- The configuration matches multiple classes: there is no discriminating characteristic among the classes. User interaction is required to resolve this ambiguity.
- The configuration does not match any class: the object is classified as "unknown". User interaction is required to classify the object.

The system displays the results by highlighting objects that do not belong to a unique class. The user can then label the object manually; by doing so, he does not only solve the problem of this object, but he also modifies the model, allowing thus other objects to be reclassified according to this incremental learning.

User interactions modify the model in a threefold manner:

- Extension: when a previously unknown configuration is classified by the user, it becomes a pattern. The model is therefore extended with new knowledge.
- Specialization: when a configuration is classified incorrectly or ambiguously by the system and the user solves the conflict, a characteristic is added to the set of pertinent characteristics. This results in a more specialized model.
- Generalization: unknown configurations are also the result of a model which is too discriminating. In this case, characteristics can be removed from the set of pertinent characteristics to render the model more general.

For a more formal description of this method please refer to [11].

2.1 The Choice of Characteristics

The pertinent characteristics for a class are determined by *2(CREM)*, during the creation or specialization step. In both situations, the choice of the characteristics affects the quality of the model.

A good set of characteristics for a class is characterized by two criteria:

- *Discrimination*: let D be the number of elements not belonging to the class and having the same value as class pattern for the evaluated characteristics set. A good characteristics set is a set in which D is kept small.
- *Homogeneity*: let H be the number of different values inside class members for the valued characteristic set. A good set of characteristics, from the point of view of homogeneity, is a set for which H is kept small.

In the first learning steps, we have little information and cannot guarantee the choice of good characteristics. If the chosen characteristic is not good from the homogeneity point of view for one of the classes, then the members of the classes will be rarely recognized. At this point the class must be extended by adding new patterns.

One solution would be to try all the combinations of the characteristics and to choose those minimizing the number of patterns but this leads to exponential growth. Therefore a greedy algorithm is used to determine a set of characteristics for each class according to its patterns, its members and the characteristics not belonging to it.

2.2 Example

We will illustrate *2(CREM)* on a simplified document classification example shown figure 3.

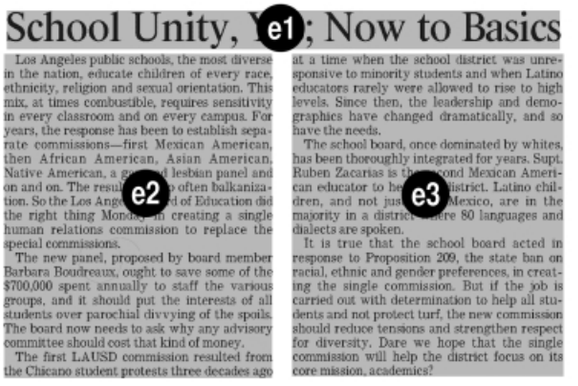

Fig. 3. Document example with three objects.

The static part of the chosen model is defined by two attributes that are the type and the font of the objects. Every object is represented by a list of "attribute-value" pairs that correspond to its configuration:

- $e_1 = \{(\text{Type} : \text{textblock}), (\text{Font} : \text{big})\}$
- $e_2 = \{(\text{Type} : \text{textblock}), (\text{Font} : \text{small})\}$
- $e_3 = \{(\text{Type} : \text{textblock}), (\text{Font} : \text{small})\}$

As we can notice, both e_2 and e_3 configurations are equivalent. Now we simulate the interactive classification process.

Only the static part of our model contains data (staticpart = $\{\text{Type}, \text{Font}\}$). No information about any classes is available yet, since there are no discriminating attributes and patterns.

At the beginning the system finds no class for any object of the document.

Step 1

- The user classifies interactively object e_1 as "title".
- The system updates the dynamic part of its model :
  - class "title"
    - $\text{pattern}_1 = \{(\text{Type} : \text{textblock}), (\text{Font} : \text{big})\}$
    - discriminating attributes = 0
- The configuration of object e_1 has become $pattern_1$ of the "title" class.

As there is a discriminating attribute in the model for the class "title", the system classifies every object in the class "title".

Step 2

- The user classifies object e_2 in the class "paragraph".
- The system updates the model's dynamic part:
  - class "title"
    - $\text{pattern}_1 = \{(\text{Type} : \text{textblock}), (\text{Font} : \text{big})\}$
    - discriminating attributes = Font
  - class "paragraph"
    - $\text{pattern}_1 = \{(\text{Type} : \text{textblock}), (\text{Font} : \text{small})\}$
    - discriminatingattributes = $\{\text{Font}\}$
- The only potential discriminating attribute was "Font" and the configuration has become pattern_1 of the class "paragraph".

Object e_1 is classified "title" and objects e_2 and e_3 are classified "paragraph".

3 Tests & Results

The first prototype of *2(CREM)* has been tested on four different applications: thread recognition, frame recognition, text lines merging into blocks and logical labeling.

All the experiments described were conducted on a set of pages from the Los Angeles Times (LAT) newspaper (see figure 4) using the XMIllum environment [8]. All the information (of the document as well as of the model) is represented in XML. The tool allows the user to browse throw the recognition results, to correct them manually and thus to adapt the document model incrementally.

The formal description of the four applications is based on graphs; please refer to [11] for more details.

Fig. 4. Page sample of LAT.

3.1 *2(CREM)* Applied to Segmentation

Thread Recognition. Concerning thread recognition, we distinguish three types of threads: thread internal separator, thread external separator and thread underline. The first one is used inside articles (often for putting eye-catchers), the second one is used to delimitate articles, and the third one is used to indicate a title or a subject (figure 5).

Fig. 5. Examples of thread types: (1), (2) and (3) are internal separators while (4) is an external separator.

The results of the thread recognition on 81 pages of the LAT are shown in table 1.

Table 1. Thread recognition results.

Thread	External Separator	Internal Separator	Underline
Recognized	90%	81%	90%
Misclassified	1%	1%	5%
Unknown	9%	18%	5%
Conflicts	0%	0%	0%

The "unknown" entry represents objects that have not been assigned to any class; the "conflicts" entry corresponds to objects assigned to more than one class. Finally, objects that were classified wrong are reported in the row "misclassified".

Frame Recognition. For the frames, we have identified two main classes which a frame may belong to:

- external separator frames separate articles from each other;
- internal separator frames are found inside articles.

The results of the frame recognition process applied on 114 pages of the LAT are shown in table 2.

Table 2. Frame recognition results.

Frames	External Separator	Internal Separator
Recognized	98%	83%
Misclassified	1%	15%
Unknown	0%	2%
Conflicts	2%	0%

Line Merging. For the line merging application, the objects to be classified are couples of neighboring text lines relatively to a vertical axis and the classes are "merging" or "not merging". Using incremental learning the system builds the whole set of possible configurations. *2(CREM)* outperforms the traditional line merging algorithms based on fixed rules.

On 29 pages of the LAT the following results have been obtained for line merging:

The presentation of the results is a little bit different, because instead of counting the number of couples of lines correctly classified, we counted the number of blocks correctly segmented. We reached 99.2% of success with only 6 incorrectly segmented blocks out of 752 blocks. These mistakes are configurations not faced during the training phase.

3.2 *2(CREM)* Applied to Logical Structure

2(CREM) has also been applied to the logical structure recognition problem and more precisely to logical labeling of text blocks. In complex documents, the diversity of labels is important. We have classified text blocks in 14 different classes: basic text, title, subtitle, author, author's function, summary, source, etc.

Table 3 shows the results for the logical structure recognition. Due to the number of classes, only the most relevant ones have been represented in our results: basic text, title, authors, summary, and source. The results of the other classes have been grouped under column "others".

Table 3. Logical structure recognition results.

Logical labeling	Basic text	Titles	Authors	Summary	Sources	Others
Recognized	90%	69%	96%	84%	94%	75%
Misclassified	0%	7%	3%	2%	3%	2%
Unknown	5%	18%	1%	0%	3%	15%
Conflicts	5%	6%	0%	14%	0%	8%

3.3 Convergence of the Model

At the beginning, no object is recognized and we would like the model to recognize almost all objects presented after the learning step. We can say that our method converges if for a finite set of objects and corrections it succeeds to build a model capable to classify correctly all set's objects. This is illustrated in figure 6.

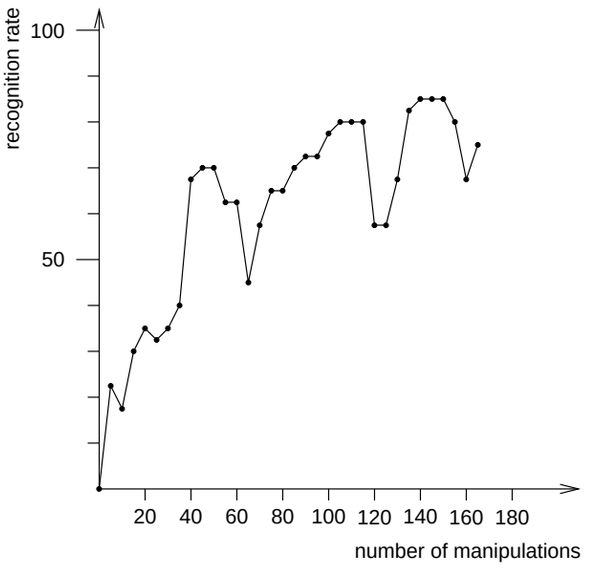

Fig. 6. Recognition rate vs. the number of manipulations.

From our observation, we believe that if the choice of the characteristics of the static part is pertinent, then the method converges; because if two objects do not belong to the same class, the system should find a characteristic which can be used to distinguish them.

In practice, the method does not converge if we can't find characteristics allowing to distinguish objects from different classes or if we cannot try all the subsets of characteristics for each class.

All the applications developed above have required only a few manipulations in order to reach an acceptable recognition rate.

4 Conclusion

In this paper we describe a classification method featuring interactive incremental learning named *2(CREM)*. Encouraging experimental results are reported concerning its application for recognition of complex structured documents.

Collecting ground-truthed documents has become a major concern of our research community. We believe that *2(CREM)* can be used successfully as a tool to build ground-truthed repositories: instead of manually labeling documents, users can, through some mouse clicks, easily produce ground-truthed data.

Several improvements are required to make *2(CREM)* more reliable and practicable. At the present stage, there is a total trust in the user actions, so that if there is a mistake made by the user *2(CREM)* blindly accepts it. Our future work on *2(CREM)* will focus on the stability of models and will handle erroneous user interactions. We would also like to test this method with other types of documents, and maybe even other applications. Another important issue is the choice of the pertinent characteristics. Finally, it might be possible to perform better by extending the models with statistical information.

References

[1] Robert M. Haralick. Document Image Understanding: Geometric and Logical Layout. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 385–390, 1994.

[2] A. Antonacopoulos B.Gatos, S.L. Mantzaris. First International Newspaper Segmentation Contest. In *ICDAR'2001: Sixth International Conference on Document Analysis and Recognition*, pages 1190–1194, Seattle, USA, September 2001.

[3] B. Gatos, S. L. Mantzaris, K. V. Chandrios, A. Tsigris, and S. J. Perantonis. Integrated Algorithms for Newspaper Page Decomposition and Article Tracking. In *ICDAR'99: Fifth International Conference on Document Analysis and Recognition*, pages 559–562, Bangalore, India, September 1999.

[4] Karim Hadjar, Oliver Hitz, and Rolf Ingold. Newspaper Page Decomposition Using a Split and Merge Approach. In *ICDAR'2001: Sixth International Conference on Document Analysis and Recognition*, pages 1186–1189, Seattle, USA, September 2001.

[5] Pierre Heroux, Eric Trupin, and Yves Lecourtier. Modélisation et classification pour la rétroconversion des documents. In *CIFED'2000: Colloque International Francophone sur l'Ecrit et le Document*, pages 413–421, Lyon, France, jul 2000.

[6] J. Hu, R. Kashi, D. Lopresti, G. Nagy, and G. Wilfong. Why Table Ground Truthing is Hard. In *ICDAR'2001: Sixth International Conference on Document Analysis and Recognition*, pages 129–133, Seattle, USA, September 2001.

[7] Frédéric Bapst. *Reconnaissance de documents assisté: architecture logicielle et intégration de savoir-faire*. PhD thesis, University of Fribourg, 1998.

[8] Oliver Hitz, Lyse Robadey, and Rolf Ingold. An Architecture for Editing Document Recognition Results Using XML. In *DAS'2000: 4th International Workshop on Document Analysis Systems*, pages 385–396, Rio de Janeiro, Brazil, December 2000.

[9] Rolf Brugger, Abdelwahab Zramdini, and Rolf Ingold. Modeling Documents for Structure Recognition Using Generalized N-Grams. In *ICDAR'97: Fourth International Conference on Document Analysis and Recognition*, pages 56–61, Ulm, Germany, August 1997.

[10] Abdelwahab Zramdini. *Study of Optical Font Recognition Based on Global Typographical Features.* PhD thesis, University of Fribourg, 1995.

[11] Lyse Robadey. *Une méthode de reconnaissance structurelle de documents complexes basée sur des patterns bidimensionnels.* PhD thesis, University of Fribourg, 2001.