Ground-Truth Production and Benchmarking Scenarios Creation with DocMining

Eric Clavier, Pierre Héroux, Joël Gardes, Eric Trupin

Ground-Truth Production and Benchmarking Scenarios Creation With DocMining

Eric Clavier[1], Pierre Heroux[2], Joel Gardes[1], Eric Trupin[2]
[1] *FTR&D,FTR&D Lannion 2 Avenue Pierre Marzin, 22307 Lannion CEDEX, France, {rntl.ball001,gardes}@rd.francetelecom.com*
[2] *PSI Laboratory, University of Rouen, 76821 Mont Saint Aignan, France, {Pierre.Heroux, Eric.Trupin}@univ-rouen.fr*

Abstract

In this paper we present the DocMining platform and its application to ground-truth datasets production and page segmentation evaluation. DocMining is a highly modular framework dedicated to document interpretation where document processing tasks are modelized with scenarios. We present here two scenarios which use PDF documents, found on the web or produced from XML files, as basis of the ground-truth dataset.

1. Introduction

Algorithm performance evaluation has become a major challenge for document analysis systems. In order to choose the right algorithm according to the domain or to tune algorithm parameters, users must have evaluation scenarios at their disposal. But efficient performance evaluation can only be achieved with a representative ground-truth dataset. Therefore users must have the possibility to create, access or modify ground-truth datasets too.

Many approaches and tools have been proposed for benchmarking page segmentation algorithm and producing ground-truth datasets. Former architectures and environment can be found in [5] [6] [8]. The ground-truth datasets are usually defined with image regions features like location, label, reading order, contained text. But information needed for a ground-truth dataset depends more on the user's evaluation intention than on a formal definition. Therefore it must be possible to upgrade a ground-truth dataset according to the evolving needs for evaluation.

Performance evaluation criteria, we found in previous works, are various and reflect those needs. Noticeable criteria are overlap ratio, regions alignment, split/merge errors.

This paper describes the DocMining platform and its application to ground-truth dataset production and performance evaluation.

This paper is organized as follow : first, we present the architecture of the DocMining platform and its major components. This platform is aimed at providing a framework for document interpretation, but its modular architecture allows multi-purpose applications based on scenarios. Then we present an application of the architecture where scenarios are designed for ground-truthing and benchmarking page segmentation algorithms.

2. The DocMining platform

The DocMining project is supported by The DocMining consortium, including four academic partners, PSI Lab (Rouen, France), Project Qgar (LORIA, Nancy, France), L3i Lab (La Rochelle, France), DIUF Lab (Fribourg, Switzerland), and one industrial partner, GRI Lab from France Telecom R&D (Lannion, France). DocMining is a multi purpose platform and is characterized by three major aspects.

At first, the DocMining architecture relies on a document-centered approach. Document processings communicate through the document itself; such an approach avoids the problems of data scattering usually met in classical document processing chains.

Second, the DocMining framework is based on a plug-in oriented architecture. Developers can conveniently add new processings, making thus the platform easily upgradeable. Document visualization and manipulation tools are also designed according to this approach, so that a user is able to fully customize the interactions with the document structure.

Third, the platform handles scenario-based operations. Running a scenario collects users' experience, which becomes part of the scenario itself. The scenario may then be transformed into a new processing corresponding to a higher-level granularity.

So the DocMining architecture is really modular because a user can create his own objects, integrate his own processings into the platform, design his own interfaces, define and run his own scenarios. In this way, the platform may be used for various interesting purposes such as benchmarking scenario creation, knowledge base creation, parameters tuning, *etc*.

2.1. Architecture overview

The platform is based on a Java/XML architecture and relies on four major components:

The PSI Library, deriving from different research works at PSI laboratory, proposes processing chains using statistical and/or structural approaches [2]. It contains a classification tools library and a XML data management library.

The Qgar software system [3] is developed by the same-named project at LORIA (www.qgar.org). It is aimed at the design of document analysis applications.

The XMillum (for XML Illuminator) platform [4] is developped by the Software, Image & Document Engineering team, at the Departement of Computer Science of the University of Fribourg. It is a general and extensible tool for the edition and visualization of all kinds of document recognition data, that are transformed into XML using XSLT stylesheets. Display and editing functionalities are delegated to plugins.

The ImTrAc package, developed by the GRI Lab at FranceTelecom R&D Lannion, provides a process engine to control processing execution and a scenario engine to control scenario execution, as well as tools for processing integration and scenario creation. An overview of the platform architecture is given in figure 1.

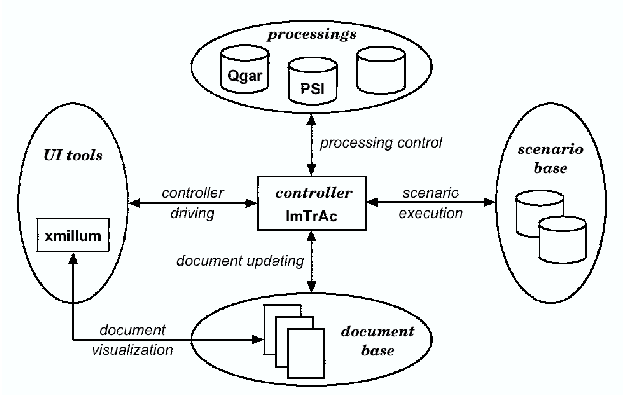

Figure 1 : DocMining platform architecture

2.2. The document structure

The DocMining architecture is based on a document centered approach. A document is represented by an XML tree built according to an XML schema. Basic elements are graphical objects defined by their type (`Binary Image`, `Connected Component`, `Text Block`, *`etc`*), their source (the document they are extracted from), and their location in the image. We did not try to build a complete predefined taxonomy of possible types: The users of the platform can define their own graphical object types when necessary. A graphical object includes intrinsic data describing the way the object is physically represented. The XML schema we have defined for that is based on basic data types such as `Freeman Chain`, `Feature Vector`, etc., but, just like previously, a user can define its own data types if necessary. However a document is more than a simple description in terms of graphical objects and data. Its structure also contains information (name, parameters, *etc*) about processings which have been applied to the document and which have provided the objects.

As shown in figure 1, objects included in the document structure are visualized with XMillum. XSLT stylesheets define what objects may be visualized, how they are visualized, and how events involving objects (e.g. mouse clicks) are handled. Each object is associated to a Java class, which performs the rendering.

2.3. Interaction between processings

As shown in figure 1, a processing has no direct access to the document structure and cannot modify it if a so-called contract, defined according to an XML schema, has not been established with the document. The contract describes the processing behavior: the way the processing modifies the XML document structure (by adding, removing or updating nodes), the kind of graphical objects it produces, and parameters that do not require access to the document structure. The objects a processing may modify or access are defined by specifying the "trigger" node (the node that enables the execution of the processing) and the "updated" nodes (the nodes which are modified by the processing).

2.4. Scenarios

In order to achieve interpretation tasks, users can interactively build scenarios, which are defined as structured combinations of document processings. There are two ways of creating a scenario. The first way is based on the contracts of the processes. As objects inputs and outputs are specified for all processings in the corresponding contracts, it is possible to determine which processings can feed a given process and then to combine processings. The other way relies on a XMillum component that we have specifically developed for the DocMining platform. It provides means to interact with the ImTrAc processing engine and to visualize the structure of the document. For each object of a document, the ImTrAc engine is able to supply the list of processings that may be applied. Once the user has chosen a processing, the engine supplies its parameters list so as to be able to launch the corresponding process. When the process terminates, the document structure is updated and the user can then interact with the newly created objects.

Each user action on the document is recorded in a scenario, which may be applied later to another document. Each step of a scenario acts as a trigger and includes an XPath expression describing the way the required objects have to be extracted from the document.

3. Page Segmentation evaluation

3.1. Obtaining the document base

PDF (Adobe® Portable Document Format) documents serve as basis for our ground-truth dataset. Indeed, the PDF format is widely used in many applications (newspaper, advertising, slides, …) and PDF documents can be easily found on the web. Moreover search engines (like google) allow to refine a search according to the document format. So it is very easy to build a PDF document base where many domains are represented.

A ground-truth dataset can also be built with newly created PDF documents. Figure 2 shows different ways to create a PDF representation of an XML document [1]. Each of the tools is freely available for download. The input XML document may be or may contain an instance of a widely used DTD such as DocBook, TEI, MathML, *etc*. Stylesheets (DSSSL or XML) are given for some of theses DTD. These can be modified so that several PDF documents with different formatting attributes may be created from a unique XML document.

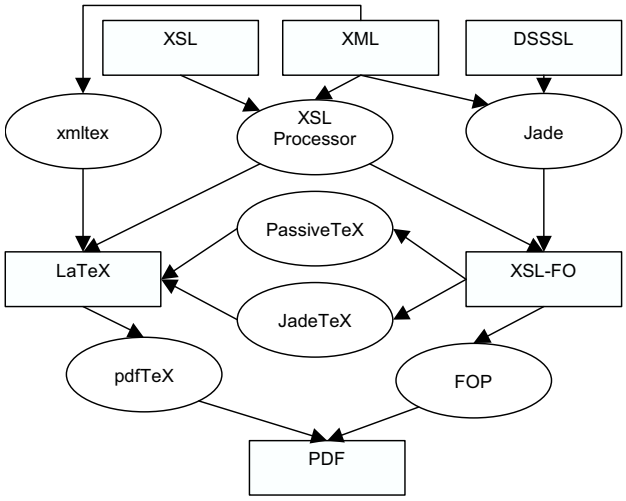

figure 2 : different ways for producing a PDF document from an XML source

With those different approaches, it is possible to build a document base which contains "real life" documents obtained through an internet search and "problem specific" documents built from an XML source.

3.2. Building the ground-truthing scenario

The main drawback of the PDF format is that it is based on a pure display approach, structural and logical information is not directly accessible, those information must be computed from the low level objects contained in the PDF document.

The ground-truth dataset is obtained through a three steps scenario:

- select the PDF documents.
- extract the physical structure from their PDF representation.
- save the generated ground-truth structure.

For each scenario step we have defined and developed a processing observing our contract approach. For example, figure 3 shows the contract we defined for the extraction of the physical structure of a PDF document. Contract noticeable elements are bold typed:

- `handled_object` : the object which is processed. Here the trigger node and the updated node are the same (see 2.3). The document must contain a `PdfDoc` element to launch the processing.
- `process_config` : the parameters of the processing
- `produced_object` : the processing produces three kind of objects (text lines, words and images) with an unknown cardinality (indicated by the list attribute).
-

```
<process_property class_name="PdfSeg">
<service name="nodeAdd">
 <handled_object>
    <object_doc type="PdfDoc"/>
    <process_config>
  <param type="ParamIn" name="ExtractImage"
support="Data" param_value="0" info="if 1
extract the images"/>
  <param type="ParamIn" name="RemoveWord"
support="Data" param_value="1" info="if 1
remove word textpieces (make the document
lighter)"/>
 </process_config>
<produced_object>
  <object_doc type="TextLine" list='yes'/>
  <object_doc type="Word" list='yes'/>
  <object_doc type="Image" list='yes'/>
</produced_object>
</handled_object>
</service>
</process_property>
```

figure 3 : contract of the PDF segmentation processing

Structure extraction from the PDF is done by using the PDF parsing API provided in the Multivalent

package [8] a java platform dedicated to the visualization of various formats documents

The resulting structure is then marshalled into an XML file observing our XML schema. Therefore, this file contains the ground-truth information of the document Figure 4 shows an excerpt of the ground truth structure and the figure 5 its visualization with XMillum.

```
<object_doc object_id="56" type="TextLine">
<object_pos h="11" w="127" x="14" y="660"/>
  <object_data>
    <ascii_data>daz, et créé ce week-end à
    </ascii_data>
  </object_data>
</object_doc>
<object_doc object_id="59" type="TextLine">
<object_pos h="11" w="127" x="14" y="670"/>
  <object_data>
    <ascii_data>Bonnefontaine dans le cadre
    des </ascii_data>
  </object_data>
</object_doc>
<object_doc object_id="67" type="TextLine">
```

figure 4 : ground-truth structure extracted from a PDF File

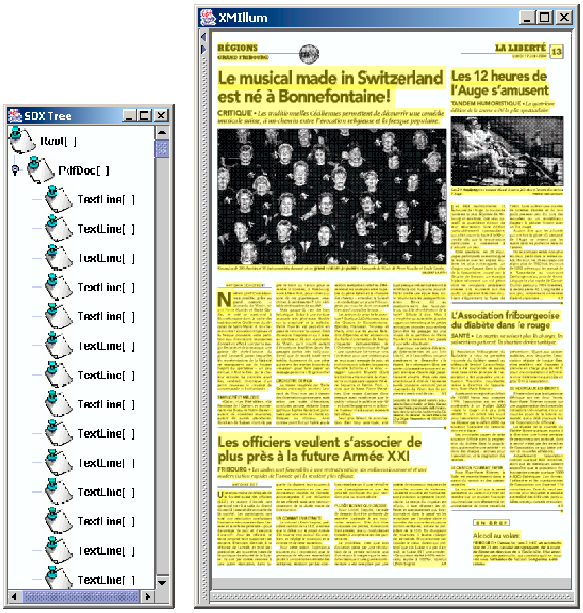

figure 5 : ground-truth structure visualization with Xmillum

Finally, our ground-truth dataset contains information concerning text lines, words (content and location) and images contained in the document. Thus we obtain a partial ground-truth that is sufficient for the first tests.

3.3. Building the benchmarking scenario

The benchmarking scenario is composed by three steps :

- Transformation of the PDF document into an image.
- Physical structure extraction using a page segmentation processing.
- Structure matching evaluation by extracting the corresponding ground-truth in the ground-truth dataset.

Transformation of the PDF document in an image is done by a processing that encapsulates a ghostscript command. At the present time, we have two segmentation algorithms, one based on a classical top down approach and the other one based on an hybrid approach[7]. Both algorithms produce a resulting XML structure which is matched with the ground-truth dataset to measure the regions overlap ratio. Ground-truth information is extracted from the dataset by using Xpath expressions which allows to select the desired corresponding structure.

Figure 6 shows the contract we defined for this node matching step. The major parameters are bold typed :

- The parameter named `KnowledgeBase` refers to the ground-truth file.
- The parameter named `XpathSelector` refers to an Xpath expression used to extract the desired objects from the ground-truth file.
- The parameter `SegmentedObjects` which is another Xpath expression, refers to the graphical objects obtained after the segmentation step.

The flexibility of Xpath expressions allows the user to select exactly what he needs, he can modify those selection expressions by choosing another kind of object (words for example) or by adding constraints (for example small areas may be filtered)

Node matching itself is done with Yanikoglu's method based on the ON pixels contained in a zone [9]. In order to ignore insignificant differences between the ground-truth regions and the segmented ones, only the black pixels content of the areas are taken into account

```
<process_property class_name="NodeMatch">
<service name="nodeAdd">
<handled_object>
  <object_doc type="BinaryImage"/>
   <process_config>
    <param type="ParamIn"
    name="KnowledgeBase" support="Data"
    param_value="gd_base.xml" info="tree
    base"/>
    <param type="ParamIn"
    name="XPathSelector" support="Data"
    param_value=
    "//object_doc[@type='TextLine']"/>
    <param type="ParamIn"
    name="SegmentedObjects"
    support="ObjectDoc" param_value=
    "object_doc[@type='TextLine']"/>
  </process_config>
```

```
<produced_object>
<object_data>
<feature name= "NodeMatching"/>
</object_data>
</produced_object>
</handled_object>
</service>
</process_property>
```

figure 6 : Contract of the Node Matching process

4. Conclusion and future works

Although many page segmentation evaluation problems are not yet addressed in this paper (blocks labeling, reading order, errors evaluation), we think that the DocMining architecture is well suited to tackle many aspects of ground-truthing and benchmarking. Indeed, DocMining's strong modularity can help building ground-truthing benchmarking scenarios according to users needs Editing and visualization tools for manipulating ground-truth datasets may be added as well. Moreover, processing modularity allows user to design their own performance evaluation algorithm. Therefore, the platform architecture allows future works to include errors evaluation processings and tools for adding new features to the datasets (labels, reading order).

In this paper, the solution we use to generate ground-truth and benchmarking scenarios for page segmentation is based on PDF documents. As shown on figure 2, PDF documents can be produced from XML data. The markup of an XML document often describes its logical structure. Therefore, a ground-truth dataset may be built with XML documents representing the logical structure and their associated PDF version corresponding to the physical structure.

5. References

[1] D. Carlisle, M. Goossens et S. Rahtz, "De XML à PDF via xmltex, XSLT et PassiveTeX" Cahiers GUTenberg n°35-36, pp. 79-114, 2000.

[2] M. Delalandre, S. Nicolas, E. Trupin and J.M. Ogier. "Symbols Recognition by Global-Local Structural Approaches, Based on the Scenarios Use, and with a XML Representation of Data", *International Conference on Document Analysis And Recognition (ICDAR)*, 2003.

[3] Ph. Dosch, K. Tombre, C. Ah Soon, G. Masini, "A complete system for the analysis of architectural drawings", *International Journal on Document Analysis and Recognition, 3(2)*, December 2000, 102-116.

[4] O. Hitz, L. Robadey, R. Ingold. An architecture for editing document recognition results using XML. Proceedings of the 4th IAPR International Workshop on Document Analysis Systems (DAS'2000), Rio de Janeiro, Brazil, December 2000.

[5] T. Kanungo, C. H. Lee, J. Czorapinski, and I. Bella. "TRUEVIZ: a groundtruth/metadata editing and visualizing toolkit for OCR". *In Proc. of SPIE Conference on Document Recognition and Retrieval,* Jan. 2001.

[6] S. Mao and T. Kanungo, "Software architecture of PSET: a page segmentation evaluation toolkit". *International Journal on Document Analysis and Recognition (4) 3*, 2002, 205-217.

[7] P.Parodi and G. Piccioli, "An efficient pre-processing of mixed-content document images for OCR systems", *Proceedings of the 13th International Conference on Pattern Recognition*, Volume: 3 , 1996, 778 -782 .

[8] T. A. Phelps and R. Wilensky, "The Multivalent Browser : A Platform for New Ideas", proc of Document Engineering 2001, Atlanta, Georgia, USA, 2001.

[9] B. A. Yanikoglu and L. Vincent, "Pink panther: a complete environment for ground-truthing and benchmarking document page segmentation,", *Pattern Recognition 31*, September 1998, 1191-1204.