

# ALDAI : Active Learning Documents Annotation Interface

Boris Chidlovskii, Jérôme Fuselier, Loïc Lecerf  
Xerox Research Centre Europe  
6, chemin de Maupertuis, F-38240 Meylan, France  
{firstname.lastname}@xrce.xerox.com

## ABSTRACT

In the framework of the *LegDoC* project at XRCE, we present a document annotation interface with an integrated active learning component. The interface is designed for automating the annotation of layout-oriented documents with semantic labels. We describe core functionalities of the interface; we pay a particular attention to the learning component, including the feature management and different strategies of deploying the active learner in the document annotation.

## 1. INTRODUCTION

The large majority of documents are created for humans and not machines, with various implicit assumptions and choices which are obvious for human readers, but difficult and ambiguous for computer programs. Consequently, one of important directions in document engineering addresses transforming documents into a form that eases the machine-oriented processing and reuse of documents.

The document annotation for machine-oriented processing ranges from collecting metadata to the document enrichment, inner-document annotation and document transformation. The most advanced level is the *document conversion* from rendering-oriented (like PDF or HTML) toward a format-independent representations, like XML, which is the modern industry standard for data exchange across service and enterprise boundaries.

Currently, the conversion of legacy documents into dense semantic XML is performed by domain experts and remains essentially manual and expensive. Taking into consideration the frequent complexity and heterogeneity of documents, the automated and accurate conversion is hardly achievable. Nevertheless, the conversion cost can be considerably reduced by deploying different and complementary methods.

At Xerox Research Centre Europe, we are conducting the Legacy Document Conversion project [2] aimed at automating different tasks of the mass document conversion to XML. The project distinguishes among three types of document annotations. The first, basic type refers to **layout** annotations that cope with the document presentation in terms of the physical rendering of elements, their  $x$  and  $y$  positions, width, height, font, etc.. The second type refers to a more abstract, **logical structure** of documents, it expresses spatial relationships between elements in a page, such as columns, headings, paragraphs and lines. The third type of annotations is **semantic** one; it refers rather to the meaning of elements than to their appearance on a page. Semantic annotations may be of different granularity; two well-known examples are the metadata that refers

to elements that describe the whole document, like *title*, *author*, *creationDate*, etc., and **entities** that are content elements of a low granularity, like person names, index entry points, etc.

The semantic annotation assumes methods for information extraction, entity recognition and inner-document annotation in the document content. In cases where the semantic entities has a simple form, writing hand-crafted rules in the form of regular expressions can be sufficient for capturing entities in the source documents.

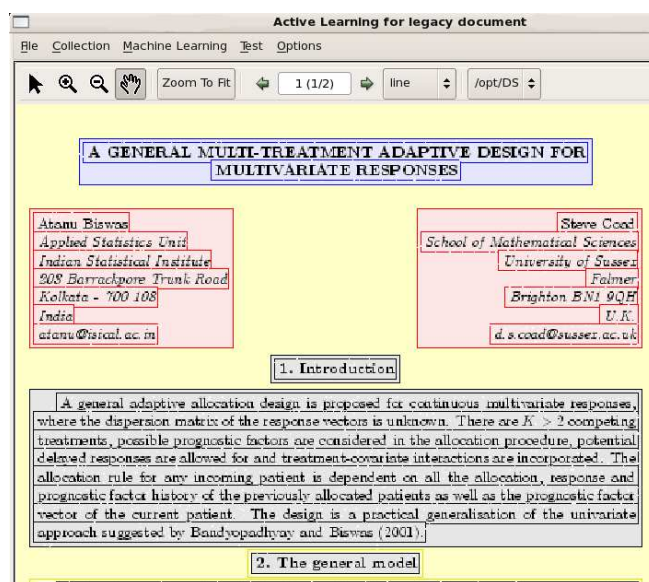


Figure 1: The WYSIWYG document annotation view.

When the manual design of annotation rules turns out to be too difficult and cumbersome, the major alternatives come from data mining and machine learning techniques. The principle of supervised learning requires an effort of annotating a subset of samples, selecting a model and training the model parameters from the available data. Unfortunately, training sets are rarely available. For cases when the annotation starts from scratch, we have developed the *Active Learning Document Annotation Interface (ALDAI)* which offers advanced functionalities to the annotator in order to smoothly and interactively progress in both the annotation process and building the accurate annotation model.

Making inner-document annotations raises certain conceptual and engineering challenges. First, the annotations may considerably vary in size, from large fragments, like multi-page sections, to low-granularity fragments, like one or few words, etc. A document fragment one may want to annotate can correspond to an XML tree

leaf, a sub-tree, etc. Second, by deploying active learning techniques, the interface allows to guide the annotation process by selecting the elements to annotate next, thus reducing the quantity of annotation samples and the risk of erroneous annotations.

## 2. ANNOTATION INTERFACE

The *ALDAI* system is composed of two major components: (1) a multi-view graphic user interface for (semantic) annotation of layout-oriented documents and (2) a learning component that learns from available annotations and suggests how to annotate unlabeled document elements.

The interface works with both HTML and layout- or logical-oriented XML documents obtained from the conversion from PDF format. The interface offers a double view on source documents, a *structural view* of the source XML files and a *WYSIWYG view* which is an application of appropriate style-sheets to documents. All annotations are referred by a unique indexing mechanism, which can be specified up to the node level, and eventually to the character level.

The WYSIWYG mode implements the *annotation highlighting* that uses bounding boxes of different colors to present semantic annotations. It also provides an *easy navigation* through pages and *zooming* of page components. Finally, it allows to visualize learner’s propositions by projecting them on the WYSIWYG document view.

All annotations are managed in the form of the document enrichment; they do not change the hierarchical structure of the document. Figure 1 shows a fragment of the annotation interface with a document being annotated, where bounding blocks represent different sub-trees in source documents while assigned semantic labels are shown with different colors.

## 3. ACTIVE LEARNING COMPONENT

The main *ALDAI* idea consists in embedding an adaptable learning component into the document annotation environment. First, by deploying active learning techniques, it allows to guide the annotation process by selecting the elements to annotate next, thus reducing the quantity of annotation samples and limiting the risk of erroneous annotations. Second, it can estimate the cost and accuracy of annotating the remaining (unlabeled) data.

The core element of the learning component is a supervised probabilistic method for inferring probabilistic classifiers for sub-trees in source documents. For each unlabeled element (which is a sub-tree)  $x$  in the source document, the classifier estimates a conditional probabilities for all labels (classes)  $y$ . It returns a probability distribution  $P(y|x)$  for all elements  $x$  in the document. According to the *maximum entropy* principle [1], we estimate the probabilities using  $P(y|x) = \exp(\Lambda \cdot F(x, y)) / Z$ , where  $F(x, y) = \{f_i(x, y)\}$  is a vector of features capturing different relationships between observations  $x$  and labels  $y$ ,  $\Lambda = \{\lambda_i\}$  is the vector of their weights trained from available data and  $Z$  is a normalization factor.

*Active learning* refers to a framework where the learning algorithm selects the instances to be labeled and then included in the training set. It often allows to significantly reduce the amount of training data needed to train a supervised learning method. Instead of annotating random instances to produce the training set, the active learning suggests to annotate those instances that are expected to maximally benefit the learning method.

We adopt the *uncertainty based sampling* principle which is based on measuring the learner confidence on unlabeled instances. According to the principle, the classifier would benefit more on labeling and including instances on which it is more uncertain when

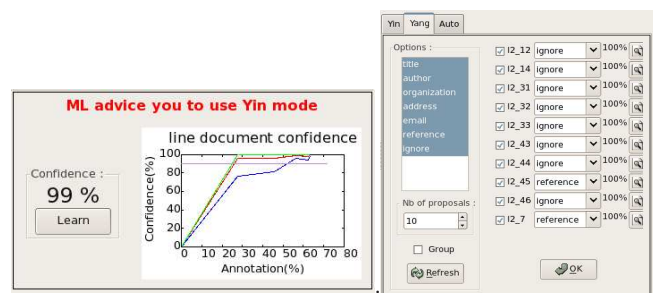


Figure 2: The confidence graph and learning panel.

attempting to classify them. Uncertainty sampling reasons using probabilities  $P(y|x)$  assigned by the classifier to every possible label  $y$  on each unlabeled element  $x$ .

**Feature management.** A careful design of features  $f(x, y)$  is critical for many annotation problems. *ALDAI* offers an extended mechanism for the feature management. By default, it offers a basic set of features extracted from layout-oriented documents [2]. In addition, it offers a high level script language to define new features or to customize existing ones. Feature  $f(x, y)$  associates label  $y$  with element  $x$  using a pair (*path, function*), where *path* is a XPath expression on the XML DOM tree that targets one or multiple nodes and *function* gets deployed through the standard Python lambda mechanism. For example, feature [RBr\_font] defined as `apply(lambda x:x.prop('font'), './following-sibling::*[1]')` assigns the value of font attribute associated with the right brother of a current node  $x$ .

**Learning modes.** To assist the user in the annotation process, the interface supports two opposite learning modes, hereafter called Yin and Yang modes. The *Yin mode* corresponds to the native active learning approach, where the model training prevails over making individual annotations. In this mode, the user invests in the model by annotating the most uncertain elements, thus increasing the accuracy of the trained model and its predictions.

In *Yang mode*, annotations prevail over training; it works with the most confident propositions, only. The associated cost is minimal as the most certain propositions are often correct and can be accepted all together (one click cost). This allows to quickly grow the annotation corpus; however, the Yang mode rarely improves the model accuracy.

The two modes are clearly complementary. It turns out that the most cost-efficient annotation strategies often go through multiple alterations of the two learning modes, switching the annotation process from the most certain to the least certain elements and back. The current interface implements a cost-based estimation algorithm that suggests at which point to make such a switch.

Finally, the *automation mode* allows to stop training and to apply the current model to all still non-annotated elements in the current document or entire collection. The uses a special window to track the confidence of the current model during the whole annotation process. Figure 2 gives examples of the learning panel and the model confidence graph.

## 4. REFERENCES

- [1] Adam L. Berger, Stephen Della Pietra, and Vincent J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.
- [2] J.-P. Chanod, B. Chidlovskii, H. Dejean, and et al. From legacy documents to xml: A conversion framework. In *Proc. European Conf. Digital Libraries*, pages 92–103, 2005.