Automated Template-Based Metadata Extraction Architecture

Paul Flynn, Li Zhou, Kurt Maly, Steven Zeil, and Mohammad Zubair

Department of Computer Science
Old Dominion University, Norfolk, VA. 23529
{pflynn,lzhou,maly,zeil,zubair}@cs.odu.edu

Abstract. This paper describes our efforts to develop a toolset and process for automated metadata extraction from large, diverse, and evolving document collections. A number of federal agencies, universities, laboratories, and companies are placing their collections online and making them searchable via metadata fields such as author, title, and publishing organization. Manually creating metadata for a large collection is an extremely time-consuming task, but is difficult to automate, particularly for collections consisting of documents with diverse layout and structure. Our automated process enables many more documents to be available online than would otherwise have been possible due to time and cost constraints. We describe our architecture and implementation and illustrate the effectiveness of the tool-set by providing experimental results on two major collections DTIC (Defense Technical Information Center) and NASA (National Aeronautics and Space Administration).

Keywords: Metadata, heterogeneous collections, automation.

1 Introduction

A number of federal agencies, universities, laboratories, and companies are placing their collections online and making them searchable via metadata fields such as author, title, and publishing organization. To enable this, every document in the collection must be catalogued using the metadata fields. A typical cataloguing process requires a human to view the document on the screen and identify the required metadata fields such as title, author, and publishing organization, and to enter these values in some online searchable database. Manually creating metadata for a large collection is an extremely time-consuming task. According to Chrystal [1], it would take about 60 employee-years to create metadata for 1 million documents. These enormous costs for manual metadata creation suggest a need for automated metadata extraction tools. The Library of Congress Cataloging Directorate recognized this problem [2] and sponsored a study, Automatic Metadata Generation Applications (AMeGA) [3], to identify challenges in automatic metadata creation.

Though time consuming, the task of identifying metadata fields by visually looking at the document is easy for a human. The visual cues in the formatting of the document along with accumulated knowledge and intelligence make it easy for a human to identify various metadata fields. Writing a computer program to automate this task is a

D.H.-L. Goh et al. (Eds.): ICADL 2007, LNCS 4822, pp. 327–336, 2007.

research challenge. Researchers in the past have shown that it is possible to write programs to extract metadata automatically for a homogeneous collection (a collection consisting of documents with a common layout and structure). Unfortunately a number of federal organizations such as DTIC [4], GPO [5], and NASA [6] manage heterogeneous collections consisting of documents with diverse layout and structure, where these programs do not work well. Furthermore, even with the best possible automated procedures, numerous sources of error exist, including some that cannot be controlled, such as scanned documents with text obscured by smudges, signatures, or stamps. A commercially viable process for metadata extraction must remain robust in the presence of these external sources of error as well as in the face of the uncertainty that accompanies any attempts to automate "intelligent" behavior. How to reach the desired accuracy and robustness for a large and evolving diverse collection consisting of documents with different layout and structure is still a major research issue. We have developed and demonstrated a novel process for extracting metadata. Among the innovations is a two-part process that directly addresses the problem of coping with large heterogeneous collections by breaking the extraction problem into smaller, manageable pieces:

- A new document is classified, assigning it to a group of documents of similar layout. The goal is to group together documents whose title or other metadata-containing pages would appear similar when viewed (by humans) from several feet away.
- Associated with each class of document layouts is a template, a scripted description of how to associate blocks of text in the layout with metadata fields. For example, a template might state that the text set in the largest type font in the top-half of the first page is, in that layout, the document title.

We have tested our process and software against the DTIC collection which contains more than one million documents and adds tens of thousands of new documents each year. The documents are diverse, including scientific articles, slides from presentations, PhD theses, (entire) conference proceedings, promotional brochures, public laws, and acts of Congress. Contributions to DTIC come from a wide variety of organizations, each with their own in-house standards for layout and format, so, even among documents of similar kind, the layouts vary widely. Our tests resulted in an overall accuracy of 83% for documents with defined templates.

2 Metadata Extraction Approaches

Existing automated metadata extraction approaches can be divided into two main categories: learning systems and rule-based systems.

Learning techniques including SVM [7][8] and HMM [9] have been employed with promising results but to relatively homogeneous document sets. The investigators' own experiments with these techniques [10] suggest a significant decline in effectiveness as the heterogeneity of the collection increases. We believe that exposure of these learning systems to heterogeneous collections tends to dilute the internal probabilities that control their internal transitions. Evolution (changing characteristics over time, such as acquiring a new source of documents in an unfamiliar format) poses a difficulty for these techniques as well, as they necessarily exhibit significant inertia resisting changes to the internally acquired "knowledge" until a significant number of examples of the new characteristics have been encountered.

Rule-based systems [11][12][13] use programmed instructions to specify how to extract the information from targeted documents. With sufficiently powerful rule languages, such techniques are, almost by definition, capable of extracting quality metadata. Heterogeneity, however, can result in complex rule sets whose creation and testing can be very time-consuming [13]. Analogies to typical software complexity metrics [14] suggest that complexity will grow much more than linearly in the number of rules, in which case even a well-trained team of rule-writers will be hard-pressed to cope with changes in an evolving heterogeneous collection and maintain a conflict-free rule set.

Our own approach [10][15] can be seen as a variant of the rule-based approach, but we finesse the complexity induced by heterogeneity and evolution by first classifying documents by layout, then providing a template for each layout, so that templates are independent of one another and individually simple.

3 Architecture and Implementation

3.1 Overview of Architecture

Our template-based metadata extraction system is composed of commercial and public domain software in addition to components developed by our team. Figure 1 shows the complete process. Documents are input into the system in the form of PDF files, which may contain either text PDF or scanned images. Some documents may contain a Report Document Page (RDP), one of several standardized forms that is inserted into the document when the document is added to the collection. For the DTIC collection, more than 50% of the documents contain RDPs offering more than 20 metadata fields.

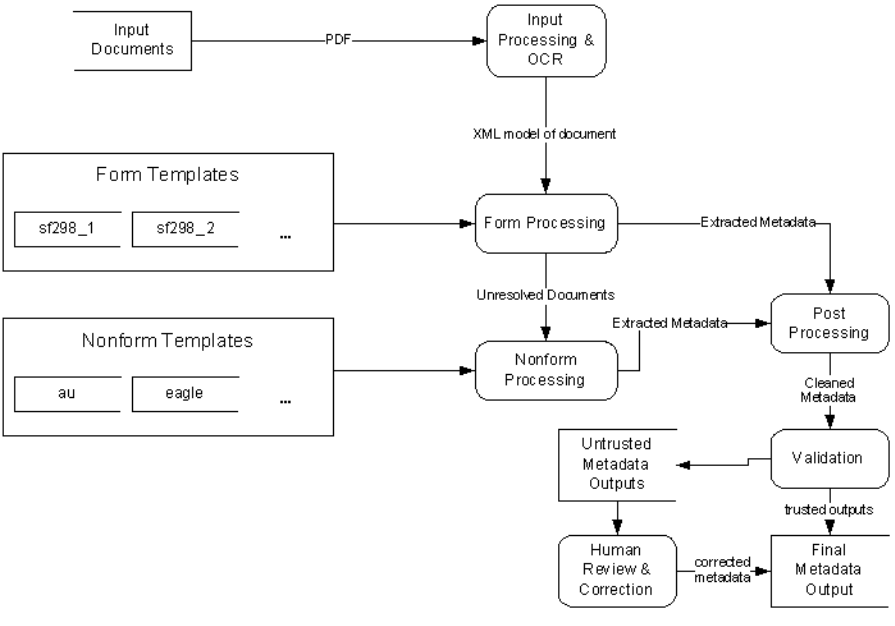

Fig. 1. Metadata Extraction Flow Diagram

The documents enter the input processing system where they are truncated, processed by an Optical Character Recognition (OCR) program and converted to a standardized XML format. The first extraction step is to search for and recognize any RDP forms present. Any documents without recognized forms enter the non-form extraction process. The non-form extraction process generates a candidate extraction solution from the templates available. After extraction, the metadata from both form and non-form processing enter the output processor. The output processor is comprised of two components: a post-processing module and a validation module. The post-processing module handles cleanup and normalization of the metadata. The final automated step of the process is the validation module which, using an array of deterministic and statistical tests, determines the acceptability of the extracted metadata. Any document that fails to meet the validation criteria is flagged for human review and correction.

3.2 Implementation

Input Processing. The source documents come into our system as PDF format files. These documents range from several pages to hundreds of pages in length. Our research into the collection has shown that the metadata we are interested in can typically be found in the first or last five pages of a document. Based on this observation, we use the program pdftk [16] to split the first and last five pages out of the document and into a new PDF document. This truncated PDF document is fed into a commercial optical character recognition (OCR) for conversion into an XML format. We have selected ScanSoft's OmniPage Pro as the OCR engine since it supports batch processing of PDF files with very good results. OmniPage saves the recognized file into a proprietary XML format which contains page layout as well as the recognized text.

The initial prototype of our extraction engine was based on the proprietary XML format used by OmniPage Pro version 14. However, by the time of the deployment of the initial prototype, the site was using OmniPage Pro version 15, which uses a different proprietary format that changed every XML tag except for the "word" tag and added dozens of new tags. Our form-based extraction engine is tightly coupled to the schema of the incoming XML documents, so supporting this new version of the OmniPage schema would require major recoding of the extraction engine, with the end result being another tight coupling to another proprietary schema. To forestall any future conflicts with schema changes, we decided to develop our own schema to decouple our project from proprietary schemas.

Independent Document Model (IDM). We based our new Independent Document Model (IDM) on the OmniPage 14 schema we already supported with our project. This step helped to minimize the re-coding cost for the extraction engine. The main structural elements are pages, regions, paragraphs, lines and words. The geometric boundaries of each of the structural elements are included as attributes. Style information such as font face, font size and font style, is recorded at the line and word levels. Alignment and line spacing are recorded at paragraph elements. Tables are composed of a sequence of cells that represent a virtual row-column table with each cell encoded with the upper-left coordinate and the row and column spans of the cell.

IDM documents are created by means of XSL 2.0 stylesheets. A different stylesheet is used for each type of source document. We have created stylesheets to support creation of IDM documents from either OmniPage 14 or 15 source documents. Similarly, additional OCR programs can be supported in the future by creation of XSL stylesheets to make the transformation.

Form Processing. Our experience with the DTIC collection has shown that about 50% of the documents contain an RDP form. The regular layout present in an RDP form makes it an attractive target for a template-based extraction process. In order to take advantage of the geometric relationships between fields in a form, we created an alternate version of our template language and extraction engine. The metadata fields are specified by a matching string and a set of rules indicating a positional relationship to one or more other fields (e.g., Figure 2). The number and layout of the fields for each different form constitute a unique signature for that form class. If a template describing form A is applied to a document containing form B, the resultant metadata returned will contain few if any fields. We have leveraged this property in the design of our extraction process.

Input processing finishes with IDM based documents exiting the input processor and entering the form processor. The processor is populated with a template developed for each version of RDP form found in the collection. We have found six different RDP forms within 9825 documents in the DTIC collection. The form processor runs the extraction process against the document using each of the templates and then selects the template, which returns the best results. If the form processor fails to match any template the document moves into the non-form extraction process described below. The extracted metadata is sent into the output processor.

```
      <field num="16->c"><line>c. THIS PAGE</line></field>
</fixed>
<extracted>
      <metadata name="ReportDate">
         <rule relation="belowof" field="1"/>
         <rule relation="aboveof" field="4|5a"/>
      </metadata>
```

Fig. 2. Form-based template fragment. The *(line)* elements in the *(field)* elements define string matching criteria. The *(rule)* elements defined for each *(metadata)* element defines the geometric placement.

Non-form Processing. As shown in Figure 1, documents without an RDP form enter the non-form processor. The documents are first transformed from IDM into another XML format called CleanML, which encodes the paragraphs and lines and their corresponding features (font size, style and alignment) into an XML structure. This simplified structure allows the extraction engine to repeatedly iterate over the content to apply the rules.

Template Construction. The non-form extraction engine also uses rule-based template extraction to locate and extract metadata. Each template contains a set of rules designed

to extract metadata from a single class of similar documents. Figure 3 shows a template example. Each desired metadata item is described by a rule set designating the beginning and the end of the metadata. The rules are limited by features detectable at the line level resolution. We hope to address this deficiency in future versions. The first step in constructing a template is to identify a set of documents which share a structural or visual similarity. Once a class is selected, the template author determines the set of rules for each metadata tag by identifying the appropriate function to select the beginning and the end of the tag.

```
<structdef pagenumber="3" templateID="arl_1">
   <CorporateAuthor>
      <begin inclusive="current">
         <stringmatch case="no" loc="beginwith">Army
             Research</stringmatch>
      </begin>
      <end inclusive="before">
         <stringmatch case="no"
            loc="beginwith">ARL</stringmatch>
      </end>
```

Fig. 3. Non-form Template fragment

```
<val:validate collection="dtic">
   <val:sum>
      <val:field name="UnclassifiedTitle">
         <val:rescale
             function="0.499 -0.01 0.5 0.5 1.0 1.0">
         <val:average>
           <val:dictionary/>
           <val:length/>
         </val:average>
         </val:rescale>
     </val:field>
```

Fig. 4. Validation script fragment for DTIC collection. Each metadata field such as "*UnclassifiedTitle*" and "*PersonalAuthor*" is assigned a function for validation.

Non-form Classification. For purposes of our discussion we define a class as a group of documents from which the metadata can be extracted using the same template. The members of a class can be selected based on structural or visual similarity. The original design of our system used several different layout classification schemes in order to separate the incoming documents into the appropriate class for extraction [10][11]. As described later, we also created a validation system to flag suspicious data extracted by a template [17][18]. We found that by applying every available template to a document, we could use the validator as a post hoc classification system for selecting the proper template. This post hoc classification system is configured by creating a "validation script" (e.g., Figure 4), which defines a set of rules to be used for calculating a confidence value for individual fields as well as an overall confidence calculation. Figure 5 is

an example of the validator output for the "alr_2" template. Table 1 shows the validation values for five of the eleven templates applied by the extraction system for the same file. (The other six templates did not produce any output for the file.) The best result, *alr_2*, differs from the next best, *alr_1*, by the extraction of an additional personal author. This is precisely the behavior and level of discrimination we desire in a classifier.

```
<metadata confidence="4.694">
   <UnclassifiedTitle confidence="0.891">Air Gun Launch
     Simulation Modeling and Finite Element Model
     Sensitivity Analysis</UnclassifiedTitle>
   <PersonalAuthor confidence="0.785">Mostafiz R.
      Chowdhury</PersonalAuthor>
   <PersonalAuthor confidence="0.713">Ala
      Tabiei</PersonalAuthor>
   <CorporateAuthor confidence="0.76">Army Research
      Laboratory Adelphi, MD 20783-1145</CorporateAuthor>
   <CorporateAuthor confidence="0.0"
      warning="CorporateAuthor: too many
        unknown words">Weapons and
      Materials Research Directorate, ARL</CorporateAuthor>
```

Fig. 5. Sample fragment of validator confidence values. In this example, we see that the second *CorporateAuthor* gives a low confidence score because of the existence of too many words not in the CorporateAuthor dictionary.

Table 1. Sample validator confidence values for a single file

Template	Total Confidence	Field Confidences			
		Unclassified Title	Personal Author	Corporate Author	Report Date
alr_2	4.694	0.891	0.785 0.713	0.760 0.000 0.546	1.000
alr_1	3.436	0.891	0.785	0.760 0.000	1.000
nsrp	1.000				1.000
rand	0.848	0.848	0.000		
nps_thesis	0.000		0.000		0.000

Output Processing. Referring back to the architecture diagram in Figure 1, the extracted metadata from both form and non-form processes enter output processing for post-processing cleanup and validation.

Post-processing. The post-processing step is designed to compensate for the inherent uncertainties involved in the OCR recognition and extraction process. We have designed a modularized post-processing system which can provide a variety of post-processing functions for each metadata field. For example, modules may be designed

to parse multiple authors from a single personal or corporate author entry and to re-format date fields into a specific standard.

As an example of a post-processing module, we have one module that attempts to standardize acceptable field values in form processing and to overcome the potential for misrecognition by the OCR software. The module analyzes specific fields by comparing the extracted data to values in an authority file. The module compares these values via fuzzy string matching based on edit distance. Additionally, the post processor can match variable phases where the comparison is successful so long as every word in the authority file entry is contained in the extracted data. We generated the authority file by extracting field data from more than 9000 documents.

Validation. The final step in our process is the validation step. The primary purpose of this step is to determine whether or not to flag the extracted metadata for human review. We will be using the same validation engine as mentioned above in post hoc classification. This validation engine uses statistical models of previously extracted metadata in the collection along with dictionaries for names and specialized content to determine the norms for the collection. While the validator will use the same validation engine to assess individual field values, we do not anticipate using the same script used in the non-form post hoc classification system. At this point we have not yet integrated the final validation module into the implementation. We are currently experimenting to determine an appropriate script to use.

4 Experimental Results

For our experiments we downloaded 9825 documents from the DTIC collection and 728 from the NASA collection. The internal distribution between forms and non-form documents for the collections are 94% RDP forms for DTIC and 21% RDP for NASA. We conducted a series of experiments to evaluate the effectiveness of the extraction process.

4.1 Form Extraction Experiments

The large number of form documents involved prohibits inspecting every document during testing. As such, we randomly sampled 100 form documents from the DTIC collection distributed roughly along the same distribution of the collection. We examined

Table 2. Results for DTIC Form Extraction

Class	Samples	Recall	Precision
Citation_1	10	100%	100%
Sf298_1	30	91%	95%
Sf298_2	30	98%	99%
Sf298_3	10	68%	96%
Sf298_4	10	100%	100%
Control	10	96%	100%

each of the 100 documents and determined the accuracy of the extracted metadata. The results of this experiment are shown in Table 2. Note that the low recall found under the SF298_3 class was due to poor quality of the source documents and resulting OCR recognition.

4.2 Non-form Extraction Experiments

We conducted experiments to confirm the efficiency of the post hoc classification system and the ability to extract the metadata. To test the ability of the system to select the appropriate template for extraction, we manually classified the DTIC non-form documents into 37 separate classes with at least 5 members. We wrote templates for the 11 largest classes and tested the ability of the extractor to correctly identify the proper class. We achieved an 87% classification accuracy when compared to manual classification results.

The overall accuracy for the non-form extractor was 66% for DTIC and 64% for NASA. The lower value is mostly due to the fact that we have only written a limited number of templates. Assuming that we write all the necessary templates, we expect accuracy in the 90% range.

5 Conclusions and Future Work

We have described our two-stage approach to metadata extraction that extends previous research in metadata extraction to growing, large, and heterogeneous collections. The basic system has been implemented and applied to two major collections with near perfect for documents that contain an RDP form and approximately 65% accuracy for those without a form. Significant contributions of our approach are the post-processing and the validation concepts. In post-processing, we clean metadata via field- and collection-specific modules. In validation we first obtain a statistical model of the collection (done only once) and use this model to validate the output.

We still have to design and implement the human correction interface together with the module that will invoke human intervention based on scores obtained in the validation phase.

References

1. Crystal, A., Land, P.: Metadata and Search: Global Corporate Circle. In: DCMI 2003 Workshop, Seattle, Washington, USA (2003), http://dublincore.org/groups/corporate/Seattle/
2. Library of Congress, Bibliographic Control of Web Resources: A Library of Congress Action Plan, http://www.loc.gov/catdir/bibcontrol/actionplan.html
3. Greenburg, J., Spurgin, K., Crystal, A.: Final Report for the Automatic Metadata Generation Applications (AMeGA) Project (2005), UNC School of Information and Library Science, http://ils.unc.edu/mrc/amega/
4. Defense Technical Information Center. Public Scientific and Technical Information Network (2007), http://stinet.dtic.mil/str/index.html
5. National Aeronautics and Space Administration. NASA Technical Reports Server (2007), http://ntrs.nasa.gov/search.jsp

6. U.S. Government Printing Office. A Strategic Vision for the 21st Century. Technical report (2004)
7. Han, H., Manavoglu, E., Zha, H., Tsioutsiouliklis, K., Giles, C.L., Zhang, X.: Rule-based word clustering for document metadata extraction. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 1049–1053. Springer, Heidelberg (2006)
8. Han, H., Giles, C.L., Manavoglu, E., Zha, H., Zhang, Z., Fox, E.A.: Automatic document metadata extraction using support vector machines. In: Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries. International Conference on Digital Libraries, pp. 37–48. IEEE Computer Society Press, Washington, DC (2003)
9. Seymore, K., McCallum, A., Rosenfeld, R.: Learning hidden Markov model structure for information extraction. In: AAAI 1999. Workshop on Machine Learning for Information Extraction (1999)
10. Tang, J., Maly, K., Zeil, S., Zubair, M.: Automated Building of OAI Compliant Repository from Legacy Collection. In: ELPUB. Proceedings of the 10th International Conference on Electronic Publishing (June 2006)
11. Mao, S., Kim, J.W., Thoma, G.R.: A Dynamic Feature Generation System for Automated Metadata Extraction in Preservation of Digital Materials. In: Dial 2004. Proceedings of the First international Workshop on Document Image Analysis For Libraries, vol. 225, IEEE Computer Society, Los Alamitos (2004)
12. Bergmark, D.: Automatic Extraction of Reference Linking Information from Online Documents. CSTR 2000-1821 (November 2000)
13. Klink, S., Dengel, A., Kieninger, T.: Document structure analysis based on layout and textual features. In: Proc. of Fourth IAPR International Workshop on Document Analysis Systems, pp. 99–111 (2000)
14. Marciniak, J.J. (ed.): Encyclopedia of Software Engineering, pp. 131–165. John Wiley & Sons, New York (1994)
15. Tang, J.: Template-based Metadata Extraction for Heterogeneous Collections. PhD thesis, Old Dominion University (2006)
16. Steward, Sid, pdftk – the PDF toolkit (2007) http://www.accesspdf.com/pdftk/
17. Maly, K., Zeil, S., Zubair, M.: Exploiting Dynamic Validation for Document Layout Classification During Metadata Extraction (2007), http://dtic.cs.odu.edu/publications/validationreal07.doc
18. Maly, K., Zeil, S., Zubair, M., Amrou, A., Aazhar, A., Ratkal, N.: A Scriptable, Statistical Oracle for a Metadata Extraction System. In: First International Workshop on Software Test Evaluation (STEV 2007), Portland, OR (October 11/12, 2007), (to appear, 2007), http://dtic.cs.odu.edu/publications/stev07.pdf