Book ID: Spi - b1471

Book Title : Handbook of Metadata, Semantics and Ontologies

S.No	chapter No	Figure No	B/W	Color	Quality	Remarks
1	ChIV-2	F001		Color	low resolution image	please provide better quality figure
2		F002		Color	low resolution image	please provide better quality figure
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						
21						
22						
23						
24						
25						
26						
27						
28						
29						
30						
31						
32						
33						
34						
35						
36						
37						
38						
39						
40						
41						
42						
43						
44						
45						
46						
47						
48						
49						
50						
51						
52						
53						
54						
55						
56						
57						
58						
59						
60						
61						
62						
63						
64						
65						

CHAPTER IV.2

TECHNOLOGIES FOR METADATA EXTRACTION

Koraljka Golub*, Henk Muller† and Emma Tonkin‡

UKOLN — The University of Bath
BA2 7AY, Bath, United Kingdom
University of Bristol, United Kingdom
**k.golub@ukoln.ac.uk*
†henkm@cs.bris.ac.uk
‡e.tonkin@ukoln.ac.uk

The chapter examines two major themes in the area of metadata extraction — formal metadata extraction and subject metadata extraction. The focus of the chapter is on textual documents, with references provided to the multimedia. Approaches to each of the two major themes are presented and discussed, with examples from the area of document classification and metadata extraction from preprints; however, analogous methods exist for use on other types of digital object, and are referenced where relevant. Suggestions are provided in which circumstances and context of use which approaches are most suitable. The chapter concludes with the thought that, owing to recognized evaluation issues, evaluation should be a major future research question in this area.

1. Introduction

Automated metadata extraction has been a challenging research issue for several decades now, major motivation being the high cost of manual metadata creation. The interest has especially grown in the 1990s when the number of digital documents started to increase exponentially. Because of the vast amount of available documents it was recognized that established objectives of bibliographic systems could be left behind [13], and that automated means could be a solution to preserve them (p. 30).

This chapter focuses on the problems underlying automated metadata extraction. We discuss methods for metadata extraction that are applicable to various forms of digital objects, and provide examples in the area of textual documents. Two major themes in the area of metadata extraction are discussed in particular, namely formal metadata extraction and subject metadata extraction.

The chapter is structured as follows: background information with definitions and document landscape are given in Sec. 2; then, formal metadata extraction with its sources of metadata, techniques, methods and evaluation are presented and discussed in Sec. 3; further, approaches to subject metadata extraction and evaluation issues are dealt with in Sec. 4; finally, concluding remarks with issues for the future are provided in Sec. 5.

2. Background

Metadata creation involves identifying elements of a document recognized as important for the purpose of document discovery and use. Traditionally, in, for example, libraries and indexing and abstracting services, this has been referred to as cataloging or bibliographic description. Metadata creation results in metadata records (or catalog or bibliographic records), which are commonly used as document surrogates — that is, records that represent a document. Information systems that make use of metadata records do so for two major reasons: digital objects, especially videos and images, are more easily described, indexed and searched *via* an accompanying metadata record than *via* direct analysis; furthermore, metadata records are generally expected to contain a high standard of descriptive information about various facets of the document, much of which is external to the digital object itself, but is the result of interpretation.

In the literature, two types of metadata creation are often distinguished: formal, and subject. In library catalogs, formal metadata creation is prescribed by International Standard Bibliographic Description (ISBD) rules and typically involves the following elements: title, responsibilities, edition, material or type, publication, production, distribution, physical description (e.g., number of pages in a book or number of CDs issued as a unit), series, notes area, identifiers (e.g., ISBN, ISSN) and terms of availability. Subject metadata creation shares essentially similar processes to those referred to as subject classification or indexing [39] and implies determination of a document's topics and choosing terms to best represent those topics. These terms can be from a controlled vocabulary (e.g., classification schemes, thesauri, subject heading systems) or freely assigned ones. Freely assigned terms can be manually assigned but can also refer to automated full-text indexing. The debate between using the former versus the latter is more than a century old but the literature acknowledges the need for both [56].

Automated metadata creation implies that the relevant processes are conducted mechanically — human intellectual processes are replaced by, for example, statistical and computational techniques. There is an intermediate approach, which can be referred to as machine-aided subject indexing or partially-automated metadata creation. Here, metadata creation is supported by mechanical means to the greatest extent possible, while final decision is left to human intellect. For example, a system could provide an appropriately tailored form for the document, or automatically fill in available form of elements with generated metadata and highlight elements which remain incomplete.

Manual metadata creation is often considered to be an expensive process, particularly when experts must be paid to generate records instead of requesting and using user-contributed metadata. Actual time taken varies greatly by domain and purpose. For example, self-archiving of an eprint to an institutional repository has been estimated to take five and a half minutes as a median average [10], requiring input of a simple metadata record comprising author, title, publication, date, etc. Creating a full bibliographic record for an information system requiring careful analysis and thorough representation usually requires much more time and effort.

Automated metadata generation is expected to play a role in reducing the metadata generation bottleneck [46]. For example, the process of depositing digital objects is rendered longer and more interactive where metadata input is required. A portion of the metadata of relevance to a given digital object can be retrieved either directly from the object itself, or from other information sources. Several methods of automatic metadata generation exist [22], including metadata extraction and metadata harvesting. The latter, metadata harvesting, relates to the collection and reuse of existing metadata. The former relates to various subtasks such as to the harvesting of information that is intrinsic to the digital object, to the content of the object (i.e., features of the document), and information that may be extracted based on comparison of the current document to other documents that this one is related to.

2.1. *Document landscape*

In the chapter, we focus on objects from which text may be extracted; in particular, documents such as web pages and preprints, stored in file formats such as PostScript, the Portable Document Format (PDF) and plain-text.

The wider world of multimedia is out of scope of this chapter but approaches to metadata extraction exists and we briefly mention some of them. Examples include audio recordings, including multilingual spoken word content [7,8,73]; audiovisual recordings, which may be indexed *via* several approaches, including text-to-speech software, as with Hewlett-Packard's Speechbot project [67]; still images, for which two primary approaches are identified in the literature — external metadata-based systems, and content-based image retrieval (CBIR), in which image content features are themselves used as query terms [43].

Each medium poses unique challenges. For example, audiovisual material presents a progression through time, whilst still images do not; however, a still image without accompanying audio contains very little contextual information to aid the process of analysis, other than perhaps pre-existing metadata such as EXIF produced at the time of image creation. At times, accompanying text will be available — in the case of audiovisual material, captions can be embedded along with the resource. Due to the complexity of directly analyzing still images, many services such as Google make use of the image context and index via the text surrounding the images (link text, captions, etc), but others like picsearch.com use information taken directly from the image as metadata, such as dimensions, file size, file type and color information. Researchers such as Clippingdale and Fujii [12], and Lee [41] have taken a similar approach to video indexing, and hybrid or multimodal approaches that index according to a combination of video and audio information have also been proposed [1]. IBM's 2006 Marvel search service (for local installation) uses advanced content analysis techniques for labeling and analyzing data, and provides hybrid and automated metadata tagging.

3. Formal metadata extraction

A large number of facts exist about any given digital object, all of which could be described as "metadata", but the majority of which are never collected or used. In Table 1, we list a number of different types of metadata that are commonly stored to describe different types of digital objects, along with the types of digital objects for which this information may be considered relevant. Elements marked with an "**" are intrinsic to certain document types, but are not present in all documents.

Also, many documents provide neither title nor author (for example, technical and data sheets often do not provide the author's name). This implies that the effectiveness of metadata extraction, or the relevance of

Table 1. Data types and corresponding formal metadata.

Type	Name	Example
Intrinsic	Filetype/attributes	PDF, MOV, MP3 at given bitrate and encoding
Intrinsic	File size	Size of file
Intrinsic	File checksum	32-bit CRC checksum
Intrinsic	File creation date	UNIX datestamp
Intrinsic	Resource language	e.g., Video contains audio streams in English, French and Russian.
Intrinsic**	Type of document	Preprint, technical report, magazine article, journal article, MSc thesis, homework, PowerPoint presentation, poster
Intrinsic**	Title	"A Christmas Carol"
Intrinsic**	Author(s)	Charles Dickens
Intrinsic**	Affiliation or contact details of author(s)	
Intrinsic**	Date of publication	Year, (may include month, day, and time)
Intrinsic**	Page count	
Intrinsic**	First and final page numbers	
Intrinsic**	Publisher, organization and title of collection/proceedings	
Intrinsic**	Document summary	
Intrinsic**	Document index, table of contents	
Intrinsic**	Sources cited/referenced within document/ bibliography	
Extrinsic	Theme	Poverty
Extrinsic	Related documents	Some forms of metadata explicitly encode various types of relationship between document objects

metadata extraction as an approach, depends greatly on the circumstances and context of use.

3.1. *Sources of metadata*

Many approaches to metadata extraction are based on document structure [23]. Document structure involves the use of the visual grammar of pages, for example, making use of the observation that title, author(s) and affiliation(s) generally appear in content header information. In this section, we discuss where metadata can be extracted — the next section describes methods and techniques for the actual extraction.

At least five general structures may be instrumental in metadata extraction:

- **Formatting structure:** The document may have structure imposed on it in its electronic format. For example, from an HTML document one can extract a DOM tree, and find HTML tags such as <TITLE>.
- **Visual structure:** The document may have a prescribed visual structure. For example, postscript and PDF specify how text is to be laid out on a page, and this can be used to identify sections of the text.
- **Document layout:** The document may be structured following some tradition. For example, it may start with a title, then the authors, and end with a number of references.
- **Bibliographic citation analysis:** Documents that are interlinked *via* citation linking or co-authorship analysis may be analyzed *via* bibliometric methods, making available various types of information.
- **Linguistic structure:** The document will have linguistic structure that may be accessible. For example, if the document is written in English, the authors may "conclude that xxx", which gives some meaning to the words between the conclude and the full stop.

3.1.1. *Formatting structure*

Certain document types contain structural elements with relatively clear or explicit semantics. One of the potential advantages of a language like HTML that stresses document structure over a language such as Postscript that stresses document layout, is that given a document structure it is potentially feasible to mechanically infer the meaning of parts of the document.

Indeed, if HTML is used according to modern W3C recommendations, HTML is to contain only structural information, with all design information contributed to CSS. This process of divorcing design from content began in the HTML 4.0 specification [4]. Under these circumstances, a large amount of information can potentially be gained by simply inspecting the DOM tree. For example, all headers H1, H2, H3, ... can be extracted and they can be used to build a table of contents of the paper, and find titles of sections and subsections. Similarly, the HEAD section can be dissected in order to extract the title of a page, although this may not contain the title of the document.

However, given that there are multiple ways in HTML to achieve the same visual effect, the use of the tags given above is not enforced and indeed many WYSIWIG tools generate a <P class='header2'> tag rather than a H2 tag, making extraction of data from HTML pages in practice difficult. A technical report by Bergmark [5] describes the use of XHTML as an intermediate format for the processing of online documents into a structure, but concedes that, firstly, most HTML documents are "not well-formed and are therefore difficult to parse"; translation of HTML into XHTML resolves a proportion of these difficulties, but many documents cannot be parsed unambiguously into XHTML. A similar approach is proposed by Krause and Marx [38].

3.1.2. *Visual structure*

In contrast to HTML, other methods to present documents often prescribe visual structure rather than document structure. For example, both Postscript and PDF specify symbol or word locations on a page, and the document consists of a bag of symbols or words at specific locations. Document structure may be inferred from symbol locations. For example, a group of letters placed close together is likely to be a word, and a group of words placed on the same vertical position on the page may be part of a sentence in a western language.

The disadvantage of those page description languages is that there are multiple ways to present text, for example, text can be encoded in fonts with bespoke encodings; the encoding itself has no relation to the characters depicted, and it is the shape of the character which conveys the meaning. In circumstances like this it is very difficult to extract characters or words, but the visual structure itself can still be used to identify sections of a document. For example, Fig. 1 shows a (deliberately) pixelated image of the first page of a paper, and even without knowing anything about the particular characters, four sections can be highlighted that almost certainly contain text, authors, affiliation and abstract.

Indeed, it turns out that visual structure itself can provide help in extracting sections of an image of, for example, legacy documents that have been scanned in. However, it is virtually impossible to distinguish between author names above the title and author names below the title, if the length of the title and the length of the author block are roughly the same.

We have performed some experiments that show that we can extract bitmaps for the title and authors from documents that are otherwise unreadable — 3–6% of documents on average in a sample academic environment [66]. An approximately 80% degree of success is achievable using a simple

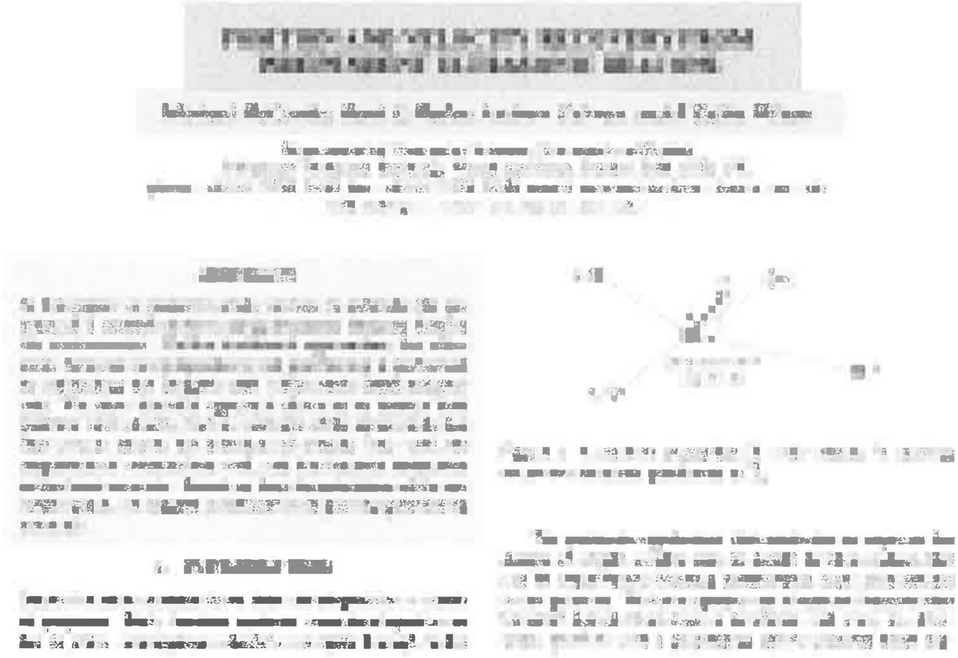

Fig. 1. Document's visual structure

image segmentation approach. These images, or indeed the entire page, may alternatively be handed to OCR software such as gOCR for translation into text and the resulting text string processed appropriately. An account of the use of appearance and geometric position of text and image blocks for document analysis and classification of PDF material may be found in [48], and a rather later description of a similar "spatial knowledge" approach applied to Postscript formatted files is given by Giuffrida, Shek, and Yang [19].

Ha *et al.* [24] note that an advantage of an approach that primarily makes use of visual features is the ease of generalization to documents in languages other than English. This approach, however, focuses solely on the problem of extracting the document title.

3.1.3. *Document layout*

From both structured description languages (such as HTML) and page description languages (such as PDF) we can usually extract the text of the document. The text itself can be analyzed to identify metadata. In particular, author names usually stand out, and so do affiliations, and even the title and journal details.

The information that can be extracted from the document structure includes:

1. Title
2. Authors
3. Affiliation
4. Email
5. URL
6. Abstract
7. Section headings (table of contents)
8. Citations
9. References
10. Figure and table captions [42]
11. Acknowledgments [18]

Extracting these purely from the document structure is difficult, but together with knowledge about words likely found in, for example, author names or titles, the extraction is feasible. A detailed discussion on the methods that we use can be found later on in this paper.

3.1.4. *Bibliographic citation analysis*

There exists a widespread enthusiasm for bibliometrics as an area, which depends heavily on citation analysis as an underlying technology. Some form of citation extraction is a prerequisite for this. As a consequence, a number of methods have been identified for this approach, making use of various degrees of automation. Harnad and Carr [28] describe the use of tools from the Open Journal Project and Cogprints that can, given well-formed and correctly specified bibliographic citations, extract and convert citations from HTML and PDF.

The nature and level of interlinking between documents is a rich source for information about the relations between them. For example, a high rate of co-citation may suggest that the subject area or theme is very similar; e.g., Hoche and Flach [29] investigated the use of co-authorship information to predict the topic of scientific papers.

The harvesting of acknowledgments has been suggested as a measure for an individual's academic impact [18], but may also carry thematic information as well as information on a social-networking level that could potentially be useful for measuring points such as conflict of interest.

3.1.5. *Linguistic structure*

Finally, the document can be analyzed linguistically, inferring meaning of parts of sentences, or relationships between metadata. For example, citations in the main text may be contained within the same sentence, indicating that the two citations are likely to be related in some way. The relation may be a positive relationship or a negative relationship, depending on the text around it, e.g., "… in contrast to work by Jones (1998), work by Thomas (1999)..."

Analyzing linguistic structure depends on knowledge of the document language, and possibly on domain knowledge. Using linguistic analysis one can attempt to extract:

1. Keywords
2. Relations between citations

3.2. *Techniques and methods*

Metadata can be extracted *via* various means, for example using support vector machines upon linguistic features [25], a variable hidden Markov model [64], or a heuristic approach [5]. Ha *et al.* [24] describe an approach that makes use of the following models upon formatting information: Perceptron with Uneven Margins, Maximum Entropy (ME), Maximum Entropy Markov Model (MEMM), Voted Perceptron Model (VP), and Conditional Random Fields (CRF). Here we will begin by discussing various approaches that are useful in metadata extraction:

- **Classification**, with the example of Bayesian classification.
- **Pattern matching**, with examples of regular expressions.
- **Direct application of observed heuristics**
- **Model fitting**: where prior knowledge is available, it may be applied as domain knowledge to build a set of models for use in metadata extraction.
- **Elicitation of grammatical structure** (ideally automated). This enables probabilistic parsing. An example is provided on the basis of Hidden Markov Models. Maximum Entropy Markov Models and conditional random fields are also discussed.

The many methods available today have different uses, competences and areas of weakness so it is likely that a complete metadata extraction tool will make use of several approaches for different tasks.

3.2.1. *Bayesian classification*

A Bayesian classifier is based on prior knowledge of statistical properties of the dataset. The prior knowledge is obtained by training the Bayesian classifier on a set of manually classified data. For example, when analyzing a set of documents where one half are cooking-recipes and the other half are newspaper clippings we may find that in cooking recipes 1% of the words was "flour", whereas in the general newspaper clippings "flour" was only 0.1%, Bayesian statistics says that if we see a document with the word "flour" in it is very likely (90%) to be a cooking recipe, and not very likely (10%) to be a general newspaper clipping.

In metadata extraction Bayesian classifiers are appropriate for use in tasks such as:

- Author name extraction
- Affiliation extraction
- Publication detail extraction

Classifiers are useful in author name extraction because names are typically very different from ordinary words found in the main body of a paper. For example, "Ruud", "John", and "Canagarajah" are very likely to be names whereas "following", "classifier" and "sentence" are very unlikely to not be names. Still, there is a small, but frequently occurring, group of words that could be either. For example, "A" is a word that occurs very frequently in English text, but it could also be an initial of a name.

However, being able to classify a string to be either a name or not a name does not necessarily mean that the Author name has been extracted. In many cases names can legitimately appear elsewhere in the text. For example, we have used the names "Bayes" and "Markov" in this chapter in the context of referring to a branch of statistics and an algorithm. Even more confusing is the string "Markov" which is a proper noun (an individual's name) that is used in the text in this chapter in reference to an algorithm. Also, in papers in Arts and Literature, many names will appear that identify the subject matter rather than the author name. Thus we conclude that a Bayesian classifier can be a good basis for a metadata extractor, but if used on its own would lead to poor results.

Bayesian classifiers are one of many approaches to classification that could be used in this context. For example, Han *et al.* [25] make use of support vector machine classification. See Sec. 4 of this chapter for discussion of various approaches to subject classification in machine learning. However, the intrinsic probabilistic nature of Bayesian classification results in both a classification and a confidence measure of the classification, making it suitable for integration into a larger system.

The strength of a Bayesian classifier is that it has a strong and simple mathematical foundation, and that it is completely problem independent. When used on text, it works as well on French as Chinese text, provided that suitable training data is available

3.2.2. *Pattern matching*

A pattern comprises a description of how to construct a sequence of characters, and by repeatedly matching the pattern over a large text, parts of the text can be classified as following the pattern. Patterns come in useful to describe entities that occur often and are more reliably identified by a pattern than by any other method.

As an example, consider the email address:

buzz.aldrin@moon.com

In order to define a pattern, we typically describe the string in terms of a regular expression. The above address could be matched with the following regular expression:

```
[^ @]*@[^ ]*
```

meaning: any sequence not comprising spaces or @-characters, followed by an @-character, followed by a sequence of characters not containing any spaces. This regular expression would also match neil.armstrong@nasa.org and it would match cakes@77 in the sentence "10 cakes@77 cents each". Patterns can be made more specific, and various languages exist to express patterns.

The weakness of patterns is that it is very difficult to build a precisely matching pattern, without missing anything out. The pattern above that matches email address will for example not match the following email "address":

{alice, bob}@malice.com

Which is a notation used on many papers to indicate that both Alice and Bob have an email address at malice.com. Although it is not hard to extend the regular expression to include spaces, it will then match much more text in front of the email address, which is undesirable. The trade-off here is one between false negatives and false positives, i.e., how often has a piece of text not been matched as an email address that should have been, and how often has a piece of text been classified as an email address that should not have been. Ideally, there are no false negatives and no false positives, but in reality it is difficult to build a system where on every 10 email addresses found at least one will not be too short or too long. This is in part a consequence of the fact that conventional notations often apply simplifications or short-cuts that are not described in a formal standard — the format of email addresses, described in RFC 2822, does not include guidance on appropriate shorthand for circumstances such as paper headers — appropriate practice is defined according to style guidelines.

Pattern matching is not limited to regular expressions. We can use a grammar to capture patterns that are more complex (and that describe, for example, the nested structure of an HTML document). A grammar comprises a set of rules, each rule describing part of the structure. For example, a simple grammar for a scientific paper might read:

```
document :: title authors affiliations abstract section* references
section :: number string newline paragraph*
paragraph :: sentence*
```

As is the case with regular expressions, it is difficult to capture all document formats using a grammar — many e-prints list affiliation per author; something that is not captured by the grammar above.

In metadata extraction patterns are typically useful to pre-classify a few bits of metadata that probably have a meaning in the metadata:

- Email addresses
- URLs
- Dates
- Numbers

We have discussed email addresses above. A URL (RFC 1738, 2396) typically starts with http:// or ftp:// and can be extracted up to the point where they have been split over multiple lines, something that happens often in citations. Dates are easily recognized because a dozen formats capture them all; the most important ones being 5/1/2008, and 5 January 2008; it is not always possible to tell which date it is (as the former could refer to either the fifth of January or the first of May, depending on the convention applied) but it is very likely to be a date. Numbers are useful to recognize because they usually refer to page numbers, volume numbers, etc.

3.2.3. *Direct application of observed heuristics*

In certain instances, the semi-structured nature of the data renders it possible to achieve reasonable accuracy in data extraction by applying a set of heuristics, rules that encompass the most commonplace structures or layouts. This is particularly common in the case of metadata extraction from HTML/XHTML documents. Here each text element is surrounded by semi-semantic markup. The aspiration is that no layout information is contained within the XHTML document itself, but within an accompanying cascading style sheet referenced by the document; however, in practice this is not always the case — and the markup used can vary greatly according to the software used to generate it. For both of these reasons, seemingly simple heuristics — such as

> *'The title tends to be at or near the top of the document and appear in a larger font than other document elements'*

can be difficult to express in terms of the simple rules or pseudocode that are often suggested, such as

> *'look for the first H1 element; if there is none, look for H2, then H3, and finally <bold>'.*

The situation is simpler when working from a plain-text document. Since there is no formatting information extant from which to work, we must rely on establishing the sequence in which elements occur. There are some conventions here that can be expressed as heuristics, such as "The title tends to appear just above the authors" names, which generally precede the "affiliation, email addresses and abstract". However, there are many valid sequences in which this information appears, depending on the template used. As a result, heuristics will provide false positives.

3.2.4. *Model fitting*

In many cases there is prior knowledge about the metadata that we want to extract, and we may wish to use this information in extracting that metadata.

For example, page numbers mostly appear at the top or bottom of the page, and each page number is usually one higher than the page number of the previous page. Similarly, citations are usually a list that contains identically formatted entries starting with one of three patterns, e.g., "[1]", "[Kirk09]", or "(Wendel and Martin, 2009)". Both use pattern matching as a first step, but then require relationships between data points in order to establish whether the extracted information is useful — in the first example there must be a numeric relationship between the matched data, in the second example there must be identical patterns used in the matching process.

Page numbers are a useful example. The pattern that we match is one of {page-boundary}[1-9][0-9]* and [1-9][0-9]*{page-boundary}, resulting in one or two matches per page of text analyzed. The model that we try and fit to this data has one unknown parameter *s* (the number of the start page) and then requires that for each page *i* ($0 = i <$ number of pages), the matched number is identical to $s + i$. For a document with *p* pages this will result in at most $2p$ possible values for *s*, and for each of those possible values the total number of candidate page numbers that agree is tallied up. If at least, say, 50% of the page numbers can be identified this way, then the start page number of this paper has been identified, and hence the end-page number can be computed.

Models like the one above need to be designed carefully — they contain many assumptions, and are therefore brittle. A first problem with the model above is acceptance of any number appearing as the first or last word on a page as a valid candidate page number, although it could be a year of publication, volume number, or indeed some numeric part of the contents that happened to be in the bottom right hand of the page. In order to filter out the years, volumes, and numerical contents we require numbers on subsequent pages to be a sequence of numbers, which will successfully isolate page numbers if the paper is more than one page long. Additionally, page numbers are often not the first or last word on a page. A title or author names may appear on the right and left hand side of the page; therefore, page numbers may be missed. The above algorithm tries to deal with that by requiring only 50% of the page numbers to agree with the model.

Even though model fitting has some weaknesses, its strength is that it allows domain knowledge to be used in extracting metadata. Template matching is a special case of model fitting. Since papers are published in only a limited number of venues — probably a few tens of thousands journals and conferences using perhaps a thousand different "style sheets" — it is possible to extract the contents of an HTML or PDF file encoded in a known template by matching it against either the visual structure or grammatical flow of content elements dictated by the template. The development by hand of a set of guidelines encompassing the majority of these "style sheets" would be an interminably long and difficult process, so techniques from machine learning are often used to solve this problem.

3.2.5. *Elicitation of grammatical structure*

3.2.5.1. Hidden Markov Models (HMMs)

The methods discussed above can determine where on a document a single word or a string of words is likely to belong. Previous examples included the problem of identifying whether a word is possibly a name, or whether a string is an email address. The Bayesian classifier assigns a probability, whereas the pattern matcher approach simply gives a binary result. In many cases, the significance of a match is determined by the context in which the classification occurs. For example, a number at the end of a citation is likely to be a publication year, and a name early on in a scientific article is likely to be an author name.

A common technique that takes context into account is the Hidden Markov Model, or HMM, described by Han *et al.* [25] as the most widely used generative learning method for representing and extracting information from sequential data. The general idea of HMM is that the problem under observation is modeled as a sequence of states. On observing an event, the state can change, and a sequence of events brings us through a sequence of states.

HMMs are therefore useful in describing systems that, although we cannot observe the underlying structure directly, generate visible patterns over time. For example, we might use a sequence of observable weather conditions to guess at the shifting state of the upper atmosphere, or model ocean currents over time by examining visible features such as sediment deposits — although these observable features are not equivalent to the hidden underlying system, they are nonetheless related.

We may describe a stream of text, by analogy, as a set of observable features that overlay a hidden underlying model, the document model. If we have removed all formatting from the document, we are left with only a long stream of text, potentially punctuated with line returns, such as:

"Confirmation-Guided Discovery of First-Order Rules PETER A. FLACH, NICOLAS LACHICHE flach@cs.bris.ac.uk lachiche@cs.bris.ac.uk Department of Computer Science, University of Bristol, United Kingdom Abstract. This paper deals with learning first-order logic rules from data lacking an..."

Underlying this is a hidden structure, a sequence of states or types; we begin at a title, give two authors' names, two email addresses, and a departmental affiliation, and then move on to the abstract.

3.2.5.2. Probabilistic parsing on the basis of a HMM

The output from the Bayesian classifier gives us a good indication as to whether a given string forms part of the title or part of a string of authors, but some terms are ambiguous, or may well appear in either context. For example, the term "Markov" may well be in a preprint title, or can be an author.

The solution to this is to create a state machine (Markov chain), in which we, as background knowledge, specify likely sequences of tokens (words). For example, a likely sequence for a scientific paper comprises a number of title tokens, followed by a newline, followed by a number of author tokens, followed by a newline, followed by a number of affiliation tokens.

Given the probabilities from the Bayesian classifier, we then examine all possible evaluations of the state machine (akin to Earley parsing [14]), and we record for each possible evaluation the likelihood of this evaluation. A very simple state-machine could be that a scientific paper is simply a Title followed by a list of Author names:

PrePrint ::= Title+ Author+

i.e., a PrePrint is one or more Title-tokens followed by one or more Author-tokens. If we apply the above grammar to the example lines on the previous page, we get 11 possible parses. These parses are (Bold denotes the title and Italics denote the author):

Confirmation-Guided Discovery *of First-Order Rules, PETER A. FLACH, NICOLAS LACHICHE*
Confirmation-Guided Discovery of First-Order *Rules, PETER A. FLACH, NICOLAS LACHICHE*
...
Confirmation-Guided Discovery of First-Order Rules, *PETER A. FLACH, NICOLAS LACHICHE*
Confirmation-Guided Discovery of First-Order Rules, PETER A. FLACH, *NICOLAS LACHICHE*

The likelihood of each parse is computed by multiplying the probability of each token belonging to the specific part of the document. For example, the likelihood of the first parse requires us to multiply the probability of "Confirmation" being a title with the probability of all other terms being authors, whereas the likelihood of the last parse requires us to multiply the probability of LACHICHE being an author with all other title-probabilities. The likelihood of the third parse shown above is the multiplication of the p_{title} values for the tokens "Confirmation ... Rules" with the p_{author} values of the tokens "PETER ... LACHICHE". Computing all likelihoods, it turns out that the likelihood for the third parse is much higher than the likelihood for all other parses, and hence, according to this metric this is the "right" parse.

In the process of multiplying, all probabilities that are less than a threshold are increased to the threshold; a zero-probability of the Bayesian classifier simply means that this token has not been seen in the training set in that circumstance; it does not imply that this token can never be seen as such.

The simple state machine above would fail if, for example, the first author had an initial "A"; or has a name that is part of the scientific literature. For this reason, in a full grammar one might cluster tokens according to visual mark-up, such as line breaks and large spaces before applying the grammar rules.

3.2.5.3. Related methods: MEMMs and CRFs

Han *et al.* [25] point out that HMMs are based on the assumption that features of the model they represent are not independent from each other, and that as a consequence HMMs have difficulty in exploiting regularities of a semi-structured real system. Hence, maximum entropy based Markov models and conditional random fields (CRFs) have been proposed to deal with independent features.

Maximum entropy based Markov models (MEMMs) are closely related to the classical HMM. The term "maximum entropy" as applied here is simply a way to state that the model is designed to be as general as possible — that is, the least biased estimate possible on the given information, the most noncommittal with regard to missing information [33].

MEMMs differ from HMMs in that MEMMs are not simply trained on examining the output tokens themselves (i.e., the word, the Bayesian classification of the word, etc) but make use of a feature set derived from them. HMMs only look at the word itself, and do not look at the large set of ways in which that word might be described. For a single word, such a feature set might include whether it is capitalized or whether it ends in a full stop, whether it contains an @ sign, whether it is a noun, etc. There is a very large number of different possible features on a word, sentence, paragraph or page level. So MEMMs may be described as taking into account "overlapping" features. For some classes of problem, particularly those with sparse datasets on which to train, using an MEMM can be expected to improve overall performance — predictably, since the features that are examined are relatively generic. The other approach mentioned, CRFs, are sometimes described as a generalization of the HMM that, whilst flexible and powerful, are relatively conceptually complex. A good introduction to CRFs is given in [68].

3.2.6. *Integrating additional evidence*

Each piece of information that has already been collected and parsed may become a useful additional source of data. For example, extracted references may be used as an additional source of information, not only about the ways in which documents are linked, but also about the documents that we have already parsed. In many cases, information that is contained within references is not contained within the document itself — for example, the publication date of the document is often absent from the document template, as is the name of the publisher, proceedings or journal, the location of the publisher or conference, and the page numbers at which the document appears. It is possible to make use of a metadata extraction database as an ever-growing knowledge base, which is able to review and correct extracted metadata as and when further pieces of evidence about the same document become available. Therefore, metadata extraction can usefully be thought of and treated as an ongoing process of iterative information discovery, gathering and reasoning, rather than a short process of information input, filtering and output of a metadata record.

3.3. *Error propagation*

Once errors in metadata exist, they propagate in various ways; reinforcing similar errors on future preprints, introduction of seemingly unrelated extra errors, and obfuscation of the data presented to the user. Firstly, a system will normally use previous classifications in order to classify future papers. In our system, paperBase, author-names, title, abstract, and classification of previous preprints are being used to predict the classification of new preprints. Once a preprint has been misclassified, future papers may be misclassified in a similar manner.

Secondly, a system typically uses the metadata found in preprints in order to establish connections between preprints. Connections can be made because two preprints are written by an author with the same name, because they cite each other, or because they cover a similar subject matter according to the keywords. Those connections can be used to, for example, disambiguate author identities. A missing link or an extraneous link would make the process of reasoning about clusters of related papers increasingly difficult.

Thirdly, the answers of search queries are diluted when errors are introduced. Cascading errors cause a disproportional dilution of search results. This is also true of user-contributed systems in which users may infer the use of classification terms through examining available exemplars.

When machine-generated classifications are provided, they are generally represented as unitary facts; either a document may be described *via* a keyword, or it may not. Consider the following example of a machine-generated classification:

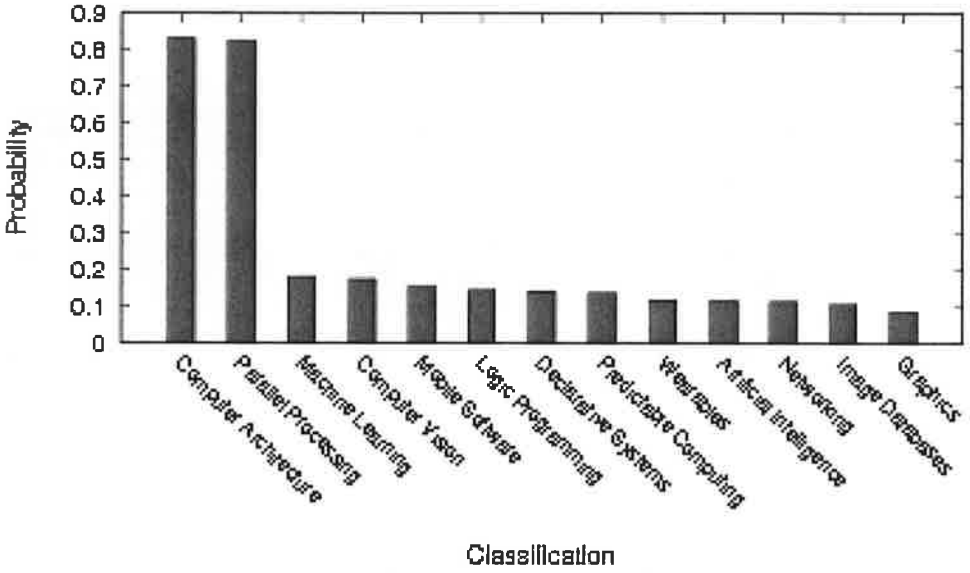

Fig. 2. A typical Bayesian classification record.

In this case, a document is considered almost certain to be about "Computer Architecture" or "Parallel Processing", and to have a diminishing likelihood of being classifiable as about "Machine Learning" or any of the other terms. In general, a threshold is placed, or the top classification accepted by default, when the result is presented, but it is this distribution that describes the paper with respect to others. The shape of this distribution is very relevant in establishing the nature and relevance of the classification. There may be no clear winner if there are many keywords with similar probability, and then our confidence in the clarity of the results may be shaken absent human evaluation of that judgment.

In the case of classifications, many options may be acceptable, but this is less the case in other situations where uncertainty exists. Consider the set of sample parses shown previously. The likelihood for the correct parse is much higher than the likelihood for all other parses. Unlike the prior example of a classification, only one of these parses can be valid. Whilst it is the most likely, we do not have total confidence in this, but we are able to generate a probability of its accuracy (our level of confidence, a value between zero and one). Hence, it is possible to provide some guidance as to the validity of this datum as a "fact" about the document.

The danger of reasoning over data in which we, or the system, have low confidence, is the risk of propagating errors. If we retain a Bayesian viewpoint, we may calculate any further conclusions on the basis of existing probabilities *via* Bayesian inference. If, however, we treat a probability as a fact and make inferences over inaccurate data without regard to degree of confidence, the result may be the production of hypotheses over which we have very little confidence indeed.

As a consequence, an extension of DC metadata to include estimates of confidence, as described in [9] is useful, as in the case of classification would be an estimate of the number of classifications considered "plausible"; the breadth or range of likely classifications, which could also be described in terms of variation or level of consistency in judgment — a similar value to that which might be generated in any other situation in which generated or contributed classifications may be treated as "votes", such as collaborative tagging systems.

If the nature and extent of the error are known, further functions that employ these values may apply this information to estimate the accuracy of the result or that of derivative functions. We note that for certain types of metadata, this problem is well-investigated. For example, author name disambiguation has received a great deal of interest in recent years, e.g., Han *et al.* [25], Han, Zha and Giles [27].

3.4. Evaluation

Once a metadata extraction system has been designed and implemented it is usually evaluated to check that it is fit for purpose. This evaluation may take a number of different forms, depending on the goal that was set. The evaluation may check that the metadata is correct — that is, little incorrect metadata is present, and/or that the metadata is complete — that is, all metadata has been filled in. Another relevant quantitative metric is the time taken to complete an extraction.

There are various methods of evaluation, some of which employ comparison against a typical manual metadata extraction process; user-studies can also be used. Many approaches depend on comparison of actual results with a set of "standard" answers — in computer science, this is often referred to as a ground truth.

The concept of a ground-truth is that there is an undisputed correct answer. Such sample data is usually created and corrected by hand. The ground truth can be used in system design, to train components of the system, like classifiers, which in some cases are trained by presentation of examples. It is also useful for measuring the performance of the system, by comparing the outcome of the metadata extraction process with the ground truth, and counting the number of errors according to an appropriate system of scoring. The ground truth is therefore often split into a training set and an evaluation set.

Since creation of a ground truth is a manual process, collecting a large ground truth is painstaking work, and often one has to compromise. One compromise is to collect a small ground truth, the other is to collect an "almost"-truth, a set of answers that is, though imperfect, fairly precise. The latter may contain typographic errors and misclassifications. This may degrade system performance if used for training. In evaluation, comparison against an incorrect ground truth will usually make the results look worse than they are. The reported error rate will comprise both human and machine errors.

Although the ground truth is a useful tool in training and evaluation it is not always the best tool for assessing the outcome. There are cases where the extracted metadata is not "right" or "wrong", but is open to interpretation. An example of this may be the subject matter of the document or in the case of very old documents it may not be established who the author of a document is, or when it was written. In this case, it is very difficult to *a priori* create a ground truth. One can ask a number of experts in the field for their input into the ground truth, and tally up votes (maybe weighted by expertise), but that does not guarantee that the ground truth is in any way complete. An extra complication is that if some of the metadata that is subjective in nature, then the interpretation of this metadata or indeed of the source of the metadata may mean that there is no fixed ground truth, but that the ground truth depends on when the document was written, or when the metadata was extracted, or when the metadata was interpreted.

To this end, another comparison method is to take the output of the metadata extraction process to a group of experts, and to ask them to assess the correctness of the data. This method has the disadvantage of requiring time from an expert panel to assess potentially a large volume of output.

The first objective metric that is important in assessing the quality of the metadata is to measure the correctness of the metadata. That is, given all the metadata that the system extracted, how much of it is actually correct metadata (compared to the ground truth or the user panel). Although it is straightforward to define when the data is correct, that is, both strings of metadata must be the same, character by character, it is much harder to define how to count errors. For example, the metadata may contain an extra space before or after the metadata, which is probably perfectly acceptable. It may contain a full stop after author initials, which is probably also acceptable. When two characters have gone missing in the title, this can be counted as one error ("the title is wrong"), two errors ("two characters are missing") or even 20 errors ("all subsequent characters are wrong").

Correct metadata is important, but not at all costs — a system can be designed to have a high degree of correctness by extracting solely metadata that the system is absolutely 100% confident about and hence extract very little information. For this reason, a second metric that is to be satisfied is that of completeness, which measures how much of the desired metadata has actually been extracted. The idea is to extract all metadata, maybe at the cost of some of the metadata being wrong or "over-complete". As an example, extracted metadata may contain both author and editor names as authors, or it may contain title and part of the abstract as a title.

The trade-off between completeness and correctness means that one usually has to allow for a few errors, and accept that some metadata is missing. One way to represent this is to plot, for a number of parameter settings, the completeness against the correctness, and to choose a parameter setting that has an acceptable error at a satisfactory level of completeness — the level where some data needs to be added, and some errors need to be corrected. This plot is known as an Receiver Operating Curve (ROC). When an ROC curve is to be made, the system has to be tested under many different parameter settings, and one must have a ground truth to mechanically check on the number of errors.

The correctness and completeness are very close to the precision and recall in information retrieval, but they are not the same. In particular, in information retrieval the answer of the query comprises a selection from a set of all possible answers. Hence, the term precision can be defined as X/A where X is the number of answers that were useful and A is the number of answers. The recall can be defined as X/C where X is the number of answers that were useful and C is all correct answers (as stipulated by the ground truth).

Completeness and correctness look at the validity of the metadata *per se.* Another aspect of the evaluation of the complete system involves user studies, studying how all people involved rate the resulting metadata. User studies can involve professionals (digital librarians and archivists), authors of papers, and users of the metadata. Pertinent points in the development of such a system include the validity of the metadata itself, and the advantages and disadvantages, both perceived and quantitatively measured, for users in various contexts such as information retrieval, resource deposit and browsing.

4. Subject metadata extraction

4.1. *Approaches to automated subject metadata extraction*

Research related to automated subject metadata extraction is spread around a number of different areas, a most obvious one being word, term or phrase extraction. Wu and Li [70] provide an overview of keyphrase extraction for different purposes, including subject metadata derivation and automated subject classification. Automated subject metadata extraction can be also seen as what has been referred to in the literature as automated subject classification, subject indexing and text categorization, to name the few terms. All these processes are often used interchangeably and have in common one aim, and that is to automatically determine topics or subjects of a document. One can distinguish between three major approaches to automated subject classification: machine learning, clustering and string matching. In this document, machine learning refers to supervised learning, and clustering to unsupervised grouping of similar documents.

Machine learning is the most widespread approach to automated subject classification. Here documents with human-assigned classes are needed because they are then used as so-called training documents based on which characteristics of subject classes are learnt. A number of different algorithms, called classifiers, are developed to this purpose. In the following step, characteristics of documents to be classified are simply compared against the characteristics of the subject classes.

The classifiers can be based on Bayesian probabilistic learning, decision tree learning, artificial neural networks, genetic algorithms or instance-based learning — for explanation of those, see, for example, Mitchell [51]. There have also been attempts of classifier committees (or metaclassifiers), in which results of a number of different classifiers are combined to decide on a class [47]. Comparisons of classifiers can be found in Schütze, Hull, and Pedersen [57], Li and Jain [45], Yang [71], and Sebastiani [58].

The basis for these processes is representation of documents as vectors of term weights. Most representative terms are chosen for each document, and non-informative terms such as stop words are removed; this process is also conducted for computing reasons and is referred to as dimensionality reduction. The term weights can be derived using a variety of heuristic principles. For example, phrases could be given higher weight than single words; bolded terms from web pages could also be given higher weight [21]. Hypertext-specific characteristics such as headings [15], anchor words [6] and metadata [17] have been experimented with. Yang, Slattery, and Ghani [72] emphasized the importance of recognizing regularities of a web page collection when choosing a heuristic principle. For example, augmenting the document to be classified with the text of its neighbors will yield good results only if the majority in the collection has the source document and the neighbors related enough.

A major problem with machine learning is that human-classified documents are often unavailable in many subject areas, for different document types or for different user groups. If one would judge by the standard Reuters Corpus Volume 1 collection [44], some 8,000 training and testing documents would be needed per class. Because of this, approaches which diminish the need for a large number of training documents have been experimented with [6, 47, 50].

A related problem is that machine-learning algorithms perform well on new documents only if they are similar enough to the training documents. The issue of document collections was pointed out by Yang [71] who showed how similar versions of one and the same document collection had a strong impact on performance.

Finally, experiments in machine learning are largely conducted under laboratory-like, controlled conditions (see Sec. 4.2). Still, examples of its application in operative information systems exist [13, 50, 65].

Clustering is another approach to automated subject classification. Here no documents with human-assigned classes are needed — instead, documents to be classified are simply compared to each other, and the ones that are similar enough are assigned the same subject and put into the same cluster of documents (hence the name of the approach).

As in machine learning, in order to allow comparison of the documents, they are first represented by vectors, which are then compared to each other using similarity measures. Here also different heuristic principles are applied to derive the vectors as to which words or terms to use, how to extract them, which weights to assign. For example, Wang and Kitsuregawa [69] improved performance by combining terms from the web page with terms from pages pointing to it and pages leading from it. Also, different similarity measures for comparing vectors can be used, a usual one being the cosine measure.

In the following step, documents are grouped into clusters using clustering algorithms. Two different types of clusters can be constructed: partitional (or flat), and hierarchical. With partitional algorithms all clusters are determined at once. A typical example is *k*-means, in which a *k* number of clusters is first randomly generated based on an initial group of documents. Then new documents are assigned to the existing clusters, resulting in new characteristics of the clusters, requiring re-computation and rearrangement of the clusters.

In hierarchical clustering, often agglomerative algorithms are used: first, each document is viewed as an individual cluster; then, the algorithm finds the most similar pair of clusters and merges them. Similarity between documents is calculated in a number of ways. For example, it can be defined as the maximum similarity between any two individuals, one from each of the two groups (single-linkage), as the minimum similarity (complete-linkage), or as the average similarity (group-average linkage) [32, 55].

Another approach to document clustering is self-organizing maps (SOMs). SOMs are a data visualization technique, based on unsupervised artificial neural networks, that transform high-dimensional data into (usually) two-dimensional representation of clusters. For a detailed overview of SOMs, see Kohonen [37].

Since in clustering (including SOMs) clusters and their labels are produced automatically, deriving the labels is a major research challenge. In an early example of automatically derived clusters [16], which were based on citation patterns, labels were assigned manually. Today a common heuristic principle is to extract between five and ten of the most frequent terms in the centroid vector, then to drop stop-words and perform stemming, and choose the term which is most frequent in all documents of the cluster. To a limited degree, relationships between clusters are also automatically derived, which is an even more difficult problem [63]. In addition, "[a]utomatically-derived structures often result in heterogeneous criteria for category membership and can be difficult to understand" [11]. Also, clusters' labels and relationships between them change as new documents are added to the collection; unstable class names and relationships are in information systems user-unfriendly, especially when used for subject browsing.

Finally, as with machine learning, experiments are largely conducted under laboratory-like, controlled conditions (see Sec. 4.2). Still, examples of its application in operative information systems exist (e.g., Clusty for clustering search-engine results, http://clusty.com/).

String matching is the third major approach to automated subject classification. Here, matching is conducted between controlled vocabulary terms and text of documents to be classified. A major advantage of this approach is that it does not require training documents (unlike machine learning), while still maintaining a pre-defined structure (unlike clustering). Also, controlled vocabularies have the additional advantage of improving precision and recall of information retrieval. Certain controlled vocabularies will also be suitable for subject browsing. This would be less the case with automatically created classes and structures of clustering or home-grown directories not created in compliance with professional principles and standards. Yet another motivation to apply controlled vocabularies in automated subject classification is to reuse the intellectual effort that has gone into creating such a controlled vocabulary [62].

This approach does share similarities with machine learning and clustering: The pre-processing of documents to be classified includes stop-words removal; stemming can be conducted; words or phrases from the text of documents to be classified are extracted and weights are assigned to them based on different heuristical principles.

A major project involving string matching was GERHARD, a robot-generated directory of web documents in Germany [52]. The controlled vocabulary used was a multilingual version of Universal Decimal Classification (UDC) in English, German and French. GERHARD's approach included advanced linguistic analysis: from captions, stop words were removed, each word was morphologically analyzed and reduced to stem; from web pages stop words were also removed and prefixes were cut off. After the linguistic analysis, phrases were extracted from the web pages and matched against the captions. The resulting set of UDC notations was ranked and weighted statistically, according to frequencies and document structure.

Online Computer Library Center's (OCLC) project Scorpion built tools for automated subject recognition, using Dewey Decimal Classification (DDC). The main idea was to treat a document to be indexed as a query against the DDC knowledge base. The results of the "search" were treated as subjects of the document. Larson [40] used this idea earlier, for books. In Scorpion, clustering was also used, for refining the result set and for further grouping of documents falling in the same DDC class [60]. Different term weights were experimented with.

In Golub [20], building on an earlier project [3], terms from the Engineering Information thesaurus and classification scheme were matched against text of documents to be classified. Plain string-matching was enhanced in several ways, including term weighting with cut-offs, exclusion of certain terms, and enrichment of the controlled vocabulary with automatically extracted terms. The final results were comparable to those of state-of-the-art machine-learning algorithms, especially for particular classes.

Other projects include Nordic WAIS/World Wide Web Project [2], Wolverhampton Web Library (WWLib) [34] and Bilingual Automatic Parallel Indexing and Classification [53].

The three above discussed approaches are applied to textual documents. Concerning (moving) images and audio documents, according to Kirkegaard [36], this research is still in its infancy, although promising results have been achieved. The automatic approach is primarily based on computational production of numerical representations of attributes [55]. Automatic approaches can also incorporate information derived from external sources [35, 55, 59]. Further analysis of non-textual documents is out of scope of this chapter.

4.2. *Evaluation*

Various measures are used to evaluate different aspects of automated subject metadata extraction and automated subject classification [71]. Effectiveness, the degree to which correct classification decisions have been made, is often evaluated using performance measures from information retrieval, such as precision (correct positives/predicted positives) and recall (correct positives/actual positives). Efficiency can also be evaluated, in terms of computing time spent on different parts of the process. There are other evaluation measures, and new are being developed such as those that take into account degrees to which a document was wrongly categorized [13, 61]. For more on evaluation measures, see Sebastiani [58].

A major problem with evaluation as it is today is that classification results are compared against existing human-assigned classes of the used document collection. Several often ignored issues are involved and discussed below.

According to ISO standard on methods for examining documents, determining their subjects, and selecting index terms [31], manual subject indexing is a process involving three steps: (1) determining subject content of a document, (2) conceptual analysis to decide which aspects of the content should be represented, and (3) translation of those concepts or aspects into a controlled vocabulary. These steps, in particular the second one, are based on a specific library's policy in respect to its document collections and user groups. Thus, when evaluating automatically assigned classes against the human-assigned ones, it is important to know the collection indexing policies. Yang [71] claims that the most serious problem in evaluations is the lack of standard document collections and shows how different versions of the same collection have a strong impact on the performance, and other versions do not.

Another problem to consider when evaluating automated classification is the fact that certain subjects are erroneously assigned. When indexing, people make errors such as those related to exhaustivity policy (too many or too few subjects become assigned), specificity of indexing (which usually means that the assigned subject is not the most specific one available), they may omit important subjects, or assign an obviously incorrect subject [39].

In addition, it has been reported that different people, whether users or professional subject indexers, would assign different subjects to the same document. Studies on inter- and intra-indexer consistency report generally low indexer consistency [54]. Markey [49] reviewed 57 indexer consistency studies and reported that consistency levels ranged from 4% to 84%, with only 18 studies showing over 50% consistency. There are two main factors that seem to affect it:

1) Higher exhaustivity and specificity of subject indexing both lead to lower consistency, i.e., indexers choose the same first term or class notation for the major subject of the document, but the consistency decreases as they choose more subjects;
2) The bigger the vocabulary, or, the more choices the indexers have, the less likely they will choose the same terms or class notations [54].

For document collections used in evaluation it is thus important to obtain indexing policies. Also, without a thorough qualitative analysis of automatically assigned classes one cannot be sure whether, for example, the classes assigned by algorithms, but not human-assigned, are actually wrong, or if they were left out by mistake or because of the indexing policy.

Today evaluation in automated classification experiments is mostly conducted under controlled conditions, ignoring the above-discussed issues. Normally it does not involve real-life situations, subject experts or users; instead, experiments typical of laboratory information retrieval tradition are applied [30]. Or, as Sebastiani [58] puts it, "… the evaluation of document classifiers is typically conducted experimentally, rather than analytically. The reason is that … we would need a formal specification of the problem that the system is trying to solve (e.g., with respect to what correctness and completeness are defined), and the central notion … that of membership of a document in a category is, due to its subjective character, inherently nonformalizable."

5. Conclusions and outlook

The effectiveness of metadata extraction — indeed, the relevance of metadata extraction as an approach — depends greatly on the circumstances and context of use. There are many scenarios in which the methods described in this document will be inappropriate, or appropriate only in terms of a partial implementation or limited use case. Choosing the best approach to subject metadata extraction and automated classification will depend on availability of a suitable controlled vocabulary (string matching), training documents (machine learning), and purpose of the application. Clustering is least suited because automatically derived cluster labels and relationships between the clusters are often incorrect, inconsistent and hard to understand. Also, clusters change as new documents are added to the collection, which is not user-friendly either.

Practical deployment of any service or application involves a great deal of evaluation, and the results are seldom generalizable to all possible usage scenarios of that software or service. Hence, the same is true of metadata extraction. However, owing to recognized evaluation issues, it is difficult to estimate to what degree subject metadata extraction of today is applicable in operative information systems. Evaluation results depend on multiple factors, such as document collection, application context, and user tasks. It is believed that evaluation methodology of automated classification where all the different factors would be included, perhaps through a triangulation of standard collection-based evaluation and user studies, should be a major further research question.

Those with an interest in metadata extraction of any flavor are likely to benefit most from active experimentation; early deployment with the ability to opt-out, beta-testing, user evaluation and agile development are all good strategies for implementing novel methods or approaches into existing services.

References

1. Albiol, A, L Torres and EJ Delp (2004). Face recognition: When audio comes to the rescue of video. In *Proceedings of the International Workshop on Image Analysis for Multimedia Interactive Services*, Lisboa, Portugal, April 2004.
2. Ardö, A, F Falcoz, T Koch, M Nielsen and M Sandfær (1994). Improving resource discovery and retrieval on the Internet: The Nordic WAIS/World Wide Web project summary report. *NORDINFO Nytt*, 17(4), 13–28.

3. Ardö, A and T Koch (1999). Automatic classification applied to the full-text Internet documents in a robot-generated subject index. In *Proceedings of the 23rd International Online Information Meeting*, London, 239–246.
4. Austin, D, D Peruvemba, S McCarron, M Ishikawa and M Birbeck (2006). XHTML™ Modularization 1.1, W3C Working Draft. Available at http://www.w3.org/TR/xhtml-modularization/xhtml-modularization.html [last date of access 30 April 2006].
5. Bergmark, D (2000). Automatic extraction of reference linking information from online documents. CSTR 2000-1821, Cornell Digital Library Research Group.
6. Blum, A and T Mitchell (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational Learning Theory*, 99–100.
7. Brown, MG, JT Foote, GJF Jones, K Sparck Jones and SJ Young (1995). Automatic content-based retrieval of broadcast news. In *Proceedings of the third ACM international conference on Multimedia*, 35–43, San Francisco, November 1995. ACM Press.
8. Byrne, W, D Doermann and M Franz (2004). Automatic recognition of spontaneous speech for access to multilingual oral history archives. *IEEE Transactions on Speech and Audio Processing*, Special Issue on Spontaneous Speech Processing, July 2004.
9. Cardinaels, K, E Duval and HJ Olivié (2006). A formal model of learning object metadata. *EC-TEL*, 74–87.
10. Carr, L and S Harnad (2005). Keystroke economy: A study of the time and effort involved in self-archiving. Unpublished public draft. Available at http://eprints.ecs.soton.ac.uk/10688/1/KeystrokeCosting-publicdraft1.pdf.
11. Chen, H and ST Dumais (2000). Bringing order to the web: Automatically categorizing search results. In *Proceedings of the ACM International Conference on Human Factors in Computing Systems*, Den Haag, 145–152.
12. Clippingdale, S and M Fujii (2003). *Face Recognition for Video Indexing: Randomization of Face Templates Improves Robustness to Facial Expression.* Lecture Notes in Computer Science, Vol. 2849/2003. Berlin / Heidelberg: Springer.
13. Dumais, ST, DD Lewis and F Sebastiani (2002). Report on the workshop on operational text classification systems (OTC-02). *ACM SIGIR Forum*, 35(2), 8–11.
14. Earley, J (1970). An efficient context-free parsing algorithm. *Communications of the Association for Computing Machinery*, 13(2), 94–102.
15. Fürnkranz, J (2002). Hyperlink ensembles: A case study in hypertext classification. *Information Fusion*, 3(4), 299–312.
16. Garfield, E, MV Malin and H Small (1975). A system for automatic classification of scientific literature. Reprinted from *Journal of the Indian Institute of Science*, 57(2), 61–74. (Reprinted in: *Essays of an Information Scientist*, 2, 356–365).

17. Ghani, R, S Slattery and Y Yang (2001). Hypertext categorization using hyperlink patterns and metadata. In *Proceedings of the 18th International Conference on Machine Learning*, 178–185.
18. Giles, CL and ID Councill (2004). Who gets acknowledged: Measuring scientific contributions through automatic acknowledgment indexing. *PNAS*, 101(51), 17599–17604.
19. Giuffrida, G, EC Shek and J Yang (2000). Knowledge-based metadata extraction from PostScript files. In *DL '00: Proceedings of the fifth ACM conference on digital libraries*, 77–84. NY, USA: ACM. DOI: http://doi.acm.org/10.1145/336597.336639
20. Golub, K (2007). Automated subject classification of textual documents in the context of web-based hierarchical browsing. Doctoral dissertation, Lund University.
21. Gövert, N, M Lalmas and N Fuhr (1999). A probabilistic description-oriented approach for categorising web documents. In *Proceedings of the Eighth International Conference on Information and Knowledge Management*, 475–482.
22. Greenberg, J (2004). Metadata extraction and harvesting: A comparison of two automatic metadata generation applications. *Journal of Internet Cataloging*, 6(4), 59–82.
23. Greenberg, J, K Spurgin and A Crystal (2006). Functionalities for automatic metadata generation applications: A survey of metadata experts' opinions. *International Journal of Metadata, Semantics and Ontologies*, 1(1), 3–20.
24. Ha, Y, H Li, Y Cao, L Teng, D Meyerzon and Q Zheng (2006). Automatic extraction of titles from general documents using machine learning. *Information Processing and Management*, 42(5), 1276–1293.
25. Han, H, CL Giles, E Manavoglu and H Zha (2003). Automatic document metadata extraction using support vector machines. In *Proceedings of the Third ACM/IEEE-CS Joint Conference on Digital Libraries*, 37–48. New York: ACM Press.
26. Han, H, CL Giles, H Zha, C Li and K Tsioutsiouliklis (2004). Two supervised learning approaches for name disambiguation in author citations. In *Proceedings of the Fourth ACM/IEEE Joint Conference on Digital Libraries*, 296–300. New York: ACM Press.
27. Han, H, H Zha and CL Giles (2005). Name disambiguation in author citations using a K-way spectral clustering method. In *Proceedings of JCDL'2005*, 334–343.
28. Harnad, S and L Carr (2000). Integrating, navigating and analysing open Eprint archives through open citation linking (the OpCit project). *Current Science*, 79(5), 629–638.
29. Hoche, S and P Flach (2006). Predicting topics of scientific papers from co-authorship graphs: A case study. In *Proceedings of the 2006 UK Workshop on Computational Intelligence (UKCI2006)*, 215–222. September 2006.

30. Ingwersen, P and K Järvelin (2005). *The Turn: Integration of Information Seeking and Retrieval in Context.* Dordrecht: Springer.
31. International Organization for Standardization (1985). *Documentation — Methods for Examining Documents, Determining their Subjects, and Selecting Index Terms: ISO 5963.* Geneva: International Organization for Standardization.
32. Jain, AK, MN Murty and PJ Flynn (1999). Data clustering: A review. *ACM Computing Surveys,* 31(3), 264–323.
33. Jaynes, ET (1957). Information theory and statistical mechanics. *Physical Review Letters,* 106, 620.
34. Jenkins, C, M Jackson, P Burden and J Wallis (1998). Automatic classification of web resources using java and Dewey Decimal Classification. *Computer Networks and ISDN Systems,* 30, 646–648.
35. Jörgensen, C (1999). Access to pictorial material: A review of current research and future prospects. *Computers and the Humanities,* 33(4), 293–318.
36. Kirkegaard, B (2008). Metadata elements preferred in searching and assessing relevance of archived television broadcast by scholars and students in media studies: Towards the design of surrogate records. Doctoral dissertation, Royal School of Library and Information Science.
37. Kohonen, T (2001). *Self-Organizing Maps,* 3rd edn. Berlin: Springer-Verlag.
38. Krause, J and J Marx (2000). Vocabulary switching and automatic metadata extraction or how to get useful information from a digital library. In *Proceedings of the First DELOS Network of Excellence Workshop on "Information Seeking, Searching and Querying in Digital Libraries".* Zurich, Switzerland.
39. Lancaster, FW (2003). *Indexing and Abstracting in Theory and Practice,* 3rd edn. London: Facet.
40. Larson, RR (1992). Experiments in automatic Library of Congress Classification. *Journal of the American Society for Information Science,* 43(2), 130–148.
41. Lee, J-H (2005). Automatic video management system using face recognition and MPEG-7 visual descriptors. *ETRI Journal,* 27(6), 806–809.
42. Liu, Y, P Mitra, CL Giles and K Bai (2006). Automatic extraction of table metadata from digital documents. In *Proceedings of the Sixth ACM/IEEE-CS Joint Conference on Digital Libraries,* 339–340.
43. Lew, MS, N Sebe, C Djeraba and R Jain (2006). Content-based multimedia information retrieval: State of the art and challenges. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP),* 2(1), 1–19.
44. Lewis, DD, Y Yang, T Rose and F Li (2004). RCV1: A new benchmark collection for text categorization research. *The Journal of Machine Learning Research,* 5, 361–397.
45. Li, YH and AK Jain (1998). Classification of text documents. *The Computer Journal,* 41(8), 537–546.

46. Liddy, ED, S Sutton, W Paik, E Allen, S Harwell, M Monsour, A Turner and J Liddy (2001). Breaking the metadata generation bottleneck: Preliminary findings. In *Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries*, 464. Roanoke, Virginia, United States.
47. Liere, R and P Tadepalli (1998). Active learning with committees: Preliminary results in comparing winnow and perceptron in text categorization. In *Proceedings of the 1st Conference on Automated Learning and Discovery*, 591–596.
48. Lovegrove, WS and DF Brailsford (1995). Document analysis of PDF files: Methods, results and implications. *Electronic publishing*, 8(2–3), 207–220.
49. Markey, K (1984). Interindexer consistency tests: A literature review and report of a test of consistency in indexing visual materials. *Library and Information Science Research*, 6, 155–177.
50. McCallum, AK, K Nigam, J Rennie and K Seymore (2000). Automating the construction of Internet portals with machine learning. *Information Retrieval Journal*, 3, 127–163.
51. Mitchell, T (1997). *Machine Learning*. New York, NY: McGraw Hill.
52. Möller, G, K-U Carstensen, B Diekmann and H Watjen (1999). Automatic classification of the WWW using the Universal Decimal Classification. In *Proceedings of the 23rd International Online Information Meeting*, 231–238, London, 7–9 December.
53. Nübel, R, C Pease, P Schmidt and D Maas (2002). Bilingual indexing for information retrieval with AUTINDEX. In *Third International Conference on Language Resources and Evaluation*, 29th, 30th and 31st May, Las Palmas de Gran Canaria (Spain), 1136–1149.
54. Olson, HA and JJ Boll (2001). *Subject Analysis in Online Catalogs*, 2nd edn. Englewood, CO: Libraries Unlimited.
55. Rasmussen, EM (1997). Indexing images. In *Annual Review of Information Science and Technology*, ME Williams (ed.), Vol. 32, pp. 169–196. Medford, NJ: Information Today.
56. Rowley, J (1994). The controlled versus natural indexing languages debate revisited: A perspective on information retrieval practice and research. *Journal of Information Science*, 20(2), 108–119.
57. Schütze, H, DA Hull and JO Pedersen (1995). A comparison of classifiers and document representations for the routing problem. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Seattle, 229–237.
58. Sebastiani, F (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1), 1–47.

59. Smeaton, AF (2004). Indexing, browsing, and searching of digital video. In Annual *Review of Information Science and Technology*, B Cronin (ed.), Vol. 38, 371–407. Medford, NJ: Information Today.
60. Subramanian, S and KE Shafer (1998). *Clustering*. OCLC Publications. Available at http://digitalarchive.oclc.org/da/ViewObject.jsp?objid=0000003409 [accessed on 10 June 2008].
61. Sun, A, E-P Lim and W-K Ng (2001). Hierarchical text classification and evaluation. In *ICDM 2001, IEEE International Conference on Data Mining*, 521–528.
62. Svenonius, E (1997). Definitional approaches in the design of classification and thesauri and their implications for retrieval and for automatic classification. In *Proceedings of the Sixth International Study Conference on Classification Research*, 12–16.
63. Svenonius, E (2000). *The Intellectual Foundations of Information Organization*. Cambridge, MA: MIT Press.
64. Takasu, A (2003). Bibliographic attribute extraction from erroneous references based on a statistical model. In *Proceedings of the Third ACM/IEEE-CS Joint Conference on Digital Libraries*, 49–60. New York: ACM Press.
65. Thunderstone (2005). Thunderstone's web site catalog. Available at http://search.thunderstone.com/texis/websearch [10 June 2008].
66. Tonkin, E and HL Muller (2008). Semi automated metadata extraction for preprints archives. *JCDL 2008*.
67. Van Thong, J-M, D Goddeau, A Litvinova, B Logan, P Moreno and M Swain (2000). SpeechBot: A speech recognition based audio indexing system for the web. *International Conference on Computer-Assisted Information Retrieval*, Recherche d'Informations Assistee par Ordinateur (RIAO), Paris, April 2000, 106–115.
68. Wallach, HM (2004). Conditional random fields: An introduction. Technical Report MS-CIS-04-21. Department of Computer and Information Science, University of Pennsylvania.
69. Wang, Y and M Kitsuregawa (2002). Evaluating contents-link coupled web page clustering for web search results. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, McLean, Virginia, USA, 499–506.
70. Wu, YB and Q Li (2008). Document keyphrases as subject metadata: Incorporating document key concepts in search results. *Information Retrieval*, 11, 229–249.
71. Yang, Y (1999). An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval*, 1(1/2), 67–88.

72. Yang, Y, S Slattery and R Ghani (2002). A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems*, 8(2–3), 219–241.
73. Ye, R, Y Yang, Z Shan, Y Liu and S Zhou (2006). Aseks: A p2p audio search engine based on keyword spotting. In *ISM '06: Proceedings of the Eighth IEEE International Symposium on Multimedia*, 615–620. Washington, DC, USA: IEEE Computer Society. Available at http://portal.acm.org/citation.cfm?id=1194217.