Metadata extraction and text categorization using Universal Resource Locator expansions

Min-Yen Kan
Department of Computer Science
School of Computing
National University of Singapore
Singapore 105001
kanmy@comp.nus.edu.sg

ABSTRACT

Uniform resource locators (URLs), which mark the address of a resource on the World Wide Web, are often human-readable and can indicate metadata about a resource. This paper explores the mining of URLs to yield categoric metadata about web resources via a three-phase pipeline of word segmentation, abbreviation expansion and classification. I apply this approach to the problem of subject metadata generation and quantify its performance relative to title- and document-based methods, both which require the retrieval of the source document.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Linguistic processing*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Language models*

Keywords

Uniform Resource Locator, metadata extraction, word segmentation, abbreviation expansion, text categorization, information content

Eligible for student paper award? No.

1. INTRODUCTION

With the level of current technology in web page indexing, automatically compiled indices only cover a fraction of the accessible web. Many of these resources are specified in terms of their *Universal Resource Locator* (URL), a string that specifies an access protocol, an internet host, and a path to the resource.

Web indexers often work by first retrieving a document and processing it. This process often yields addresses of additional documents that can be retrieved and processed in subsequent iterations. As web resources are often hyperlinked to more than a single page, this process creates a growing list of documents to be spidered – documents whose URLs are known but which have not yet been retrieved and processed. Work on focused crawling can partially help in addressing this bottleneck (e.g., [1, 2], among others). Techniques to extract information based solely on URLs can also target this bottleneck. A central hypothesis of this paper is that a lightweight process that infers metadata from the URL can help in creating a secondary information source that can be searched by users, even if the source document has yet to be examined.

Often the URL itself is quite informative and a human can glean a large amount of information from it without needing to examine the actual contents of the resource. The URL can indicate its language, subject matter, date of publication, author, or whether it is a personal homepage, a product page. Building an automated system that infers such metadata allows other applications downstream to provide more services while providing annotators default values for authority metadata.

URLs are often meant to be easily recallable by humans, and websites that follow good design techniques will encode indicative metadata about their resources in their domain naming convention as advocated by best practice guidelines ([8], pg. 26). Websites that present a large amount of expository information often break their contents into a hierarchy of pages on subtopics. This information structuring for the web often is mirrored in their URLs as well [3].

This paper reports on a system called Meurlin (Metadata Extraction from URLs) that provides a URL to faceted metadata service. Meurlin performs this service in a three-stage process. In its first phase, segmentation, Meurlin takes the URL and chunks the URL into word-like segments (e.g., *http www nytimes com* → *http www n y times com*). The expansion process follows, taking each segment and expanding it to a full-formed word when plausible (e.g., → *http www new york times com*). The final step, metadata extraction, employs machine learning on the expanded form to categorize the resource into metadata components. Figure 1 shows the architecture of the system.

In the next section, I will illustrate the types of inferences that the system attempts to find. In the sections following, I overview related work and then discuss the three stages of processing, each accompanied by evaluation. The paper offers analyses and reasons for each phase's performance before concluding.

2. URL METADATA EXTRACTION

NUS TR 10/3. October 2003.

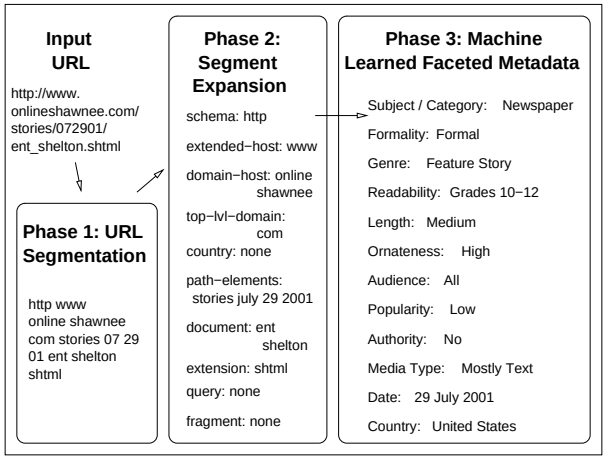

Figure 1: Proposed architecture for URL metadata extraction.

Two example URLs culled from the Open Directory Project [9], a open-source hierarchically categorized directory similar to Yahoo!, show this intuition:

1. `http://www.spart5.k12.sc.us/schools/drh/`
2. `http://www.onlineshawnee.com/stories/072901/ent_shelton.shtml`

From URL domain conventions, one might guess that the first page is about a specific school in South Carolina in the US. Furthermore, one would know that the school is a K-12 school (primary to high school; not college or university level) and that the school might be named and have "DRH" as its initials. Similarly for the second example, one might guess that the page is a feature article of the online version of a newspaper or magazine called the Shawnee, on a topic related to some named entity, Shelton, and that it was published or posted on July 29, 2001. By the date format, we might guess that this is a publication in the U.S. Additionally, one might guess that the page contains other information besides the story itself, such as a navigation menu or page counter, based on the server parsed html (.shtml) extension.

3. RELATED WORK AND BACKGROUND

As specified, the system has much in common with related work in word segmentation and abbreviation detection and expansion, as well as some recent work in automatic metadata generation.

Word segmentation techniques – for segmenting languages without spaces (e.g., Chinese) – are applicable to this task as URLs may have concatenated elements. [4] categorizes word segmentation approaches into four categories: 1) statistical methods, 2) dictionary based methods, 3) syntax-based methods and 4) conceptual methods.

Phase II – the task of expanding abbreviations – can be viewed as a special form of the word sense disambiguation (WSD) problem. It differs from WSD in that the underlying semantic meaning of an abbreviation has a different surface form, a difference that is capitalized on in this paper. In general, contextual features are used determine the word sense or the appropriate expansion of a target token. Acquiring the knowledge to properly utilize a token's context has been an open research area, in which hand-coded rules, machine readable dictionaries, supervised and unsupervised learning on text corpora [11, 14] have all been applied.

In this paper, I concentrate on statistical and dictionary based methods for performing these two tasks, as more knowledge-intensive methods (e.g., syntax and conceptual methods) do not seem as applicable in the sublanguage of URLs.

The cataloging community has long provided metadata to aid in search and retrieval of information. This process is very labor-intensive, and work has begun in the research community to provide accurate automatic metadata generation in specific domains (e.g, lesson plans [15]). The system reported here represents a complementary approach that is both domain-independent and computationally inexpensive, providing scalably-accurate faceted metadata.

The experiments that I detail in this paper use the Open Directory Project as their primary source of information. The ODP data provides category information, URL, title and short descriptions of each page in its index of over three million web-accessible resources. As the ODP data contains some formatting errors, I first preprocessed the data so that all purported resource addresses conformed to the URI protocol standard.

4. PHASE I - SEGMENTING URLS

A given URL is first segmented into its components as given by the URI protocol (e.g., *scheme :// host / path-elements / document . extension ? query # fragment*), and the internet hostname is further broken down into its domain components (e.g., five one-token components in *www . comp . nus . edu . sg*). Subsequently, four techniques were tried to further segment and expand the URLs. These are summarized below, with the more complex procedures further detailed.

1. **Non-alphanumeric character segmentation** - Segment the URL into *chunks*[1] wherever one or more non-alphanumeric characters appear (e.g., *ent_shelton* $\rightarrow$ *ent shelton*). This method is used as a baseline.
2. **Information content reduction** - This method takes the chunks after running Method 1 and attempts to split them further by using reduction in information content as a criterion.
3. **Title token based finite state transducer (FST)** - This method takes the chunks after running Method 1 and tries to further split them down by observing likely expansions in the title using a finite state framework.
4. **Cascaded** - This method uses the title-based FST method first, but employs the IC reduction method as a fallback, if no segments are suggested by the FST method.

The second method – segmentation by information content reduction relies on document frequency data compiled on the Stanford WebBase project [5]. The project provides the document frequency of over 113 billion tokens on the web.

[1]I use the term "chunks" to denote the tokens returned by this method. Any further splits of these chunks are denoted as "segments" in this paper.

Chunks coming from Method 1 are examined to see whether they can be further segmented. The algorithm examines all $2^{||c||}$ possible partitions of the chunk c and calculates the sum of the information content (IC) of each partitioning, defined as its negative log probability of occurrence, $-log(p(x))$. Because of the exponential expense of looking at long chunks, the system only attempts to expand chunks of 16 characters or less[2].

To calculate the IC for partition elements, their probability is needed. Making the assumption that a token's probability is approximated by its WebBase document frequency, I compiled probabilities for the one million most frequent tokens in the WebBase corpus. This in turn allows the system to calculate a partition element's IC.

A partitioning that has a lower IC sum than other partitionings can be said to have a lower amount of uncertainty, and is a more probable parse of the chunk. A similar approach has been applied [7] quite successfully to the word segmentation problem.

The system finds the partition with the minimal IC and compares it with the IC of the string as a series of characters (using unigram character probabilities). If the minimum scoring partition has lower information content than the chunk as a series of characters, then the chunk is further broken down into the partition's segments. Otherwise the chunk is kept as a single segment. Figure 2 summarizes this approach in pseudocode.

given chunk c in URL from Method 1 do
set minIC IC_m = $+\infty$
set bestPartition B = null
for each partition p, where $p \in P = 2^{||s||}$ do
  // results in segments $s_1 ... s_n$
  set partitionIC $IC_p = \sum_{i=1}^{n} -logPr(s_i)$
  if $IC_p < IC_m$ then update
    $B =< s_1, ... s_n >$
    $IC_m = IC_p$
  end if
end for
set unigramIC $IC_u = \sum_{i=1}^{||s||} -logPr(letter_i)$
if $IC_m < IC_u$ then return B
else return s

Figure 2: Pseudocode for information content reduction based segmentation.

The third method – segmentation by tokens in the title – employs the title as a source of information to suggest segmentations. A training corpus, consisting of URLs and their titles, is first mined for likely segmentations and expansions. For this method, I used about 50% of the available Open Directory Project's <URL, Title> tuples as the training corpus, totaling 1.6 million tuples. In the testing corpus, candidate segments that correspond to ones that were expanded in the training set are then segmented. For example, if the training corpus showed that *nytimes* could be split as *n y times* then if the same chunk is observed in testing, it will segmented in the same way.

The task of matching a chunk to title tokens is done by a weighted non-deterministic finite-state transducer. The transducer has a small set of rules that associate a score with certain moves that match or skip letters in the title tokens with corresponding letters in the chunk. The expansion must cover all letters in the chunk to be considered valid. The expansion or segmentation that scores highest is used as the expansion for a particular instance. The rules that are used are listed in Table 1.

FST Rule	Score	Output
1. Match the initial letter in the subsequent token	2	$\|l$
2. Match the initial letter in a non-subsequent token	1	$\|l$
3. Match a subsequent letter in the current token	1	l
4. Match the final letter in the current token	3	l
5. Skip a character in the candidate expansion	0	ϵ

Table 1: Rules for the title expansion weighted FST.

As an example, given the chunk *nytimes* and title tokens "New York Times", the FST selects the following series of transitions: (ø $\xrightarrow{R_1}$ N $\xrightarrow{R_5}$ e $\xrightarrow{R_5}$ w $\xrightarrow{R_1}$ Y $\xrightarrow{R_5}$ o $\xrightarrow{R_5}$ r $\xrightarrow{R_5}$ k $\xrightarrow{R_1}$ T $\xrightarrow{R_3}$ i $\xrightarrow{R_3}$ m $\xrightarrow{R_3}$ e $\xrightarrow{R_4}$ s), as it has the maximal score 12, and outputs $|n|y|times$.

Of the 1.6 million training URLs chunked by Method 1 and fed through the training component, 482,874 URLs (29%) had chunks that could be further segmented, as these segments could be mapped to multiple words in the title. Their segmentations can then be employed for segmenting new URLs. The segmentations and their associated title tokens were also stored for use in expansion, discussed in Section 5.

4.1 Evaluation

One way to assess the effectiveness of the segmentation is through extrinsic evaluation, by measuring its impact on a task. I demonstrate this later in Section 6. However, an intrinsic evaluation of the module itself is also important, which in this case requires a manual evaluation. As manual evaluation is expensive, I give only preliminary results of the accuracy of this process of a set of 100 randomly chosen URLs processed by each algorithm, shown in Table 2, along with corpus-wide segmentation statistics. Judgments were made only with the original URL and the resulting segmented one. A judgment is made per original Method 1 segment, and a correct judgment is only awarded in the case of all parts are correctly segmented (e.g., $n|y|times$ is counted as correct and $n|ytimes$ is incorrect).

The segmentation algorithms were thus run on a separate test corpus of 19,778 URLs (the selection criteria further detailed in Section 6).

Method	Expansions (Prelim. Evaluation)			# URLs further expanded
	correct	wrong	missed	(% of corpus)
1. Simple segmentation	0	0	86	N/A
2. IC reduction	31	9	46	10,519 (53.1%)
3. Title based FST	12	0	74	11,209 (56.6%)
4. Cascaded	37	9	40	11,309 (57.1%)
Total Expansions:			86	

Table 2: Statistics for further segmentation of Method 1's chunks in the 19,778 URL testing corpus, along with preliminary evaluation on the first 100 documents.

In compiling the gold standard, I encountered some problems in determining when a segment should be split. For example *frontpage* could be split – either the whole or the parts could be counted as

[2]Dynamic programming techniques, such as memoizing, can significantly decrease the running time of the algorithm from its exponential bound. At the time of the experiments, these techniques had not been implemented in the system, thus the brute force technique was used.

correct. As the purpose of the evaluation was to quantify the contribution of the algorithms, it was only necessary to have a consistent standard.

Although preliminary, the manual evaluation points towards the IC based reduction being more aggressive in decomposing chunks, and the FST based approach being more accurate. At first this does not seem to coincide with the percentages on the testing corpus. However, multiple segments can be broken within a single URL, which would not be reflected in the percentages. In the manual evaluation, using the IC reduction as a fallback measure increases recall tremendously at the expense of some precision.

Also, I had limited the IC reduction method to chunks of length 16 or less for computational reasons. In fact 1,366 URLs in the testing corpus (6.9%) had at least one unbroken segment of length 17 or greater. In the manual evaluation, nine of the 46 missed expansions would have been properly resolved (running the algorithm to completion), giving a significant boost in performance.

5. PHASE II - SEGMENT EXPANSION

The resulting segmentation now undergoes an expansion phase, in which its segments are mapped to fuller forms when plausible. However, these segments can often be expanded in more than one way. For example, the URL segment "md" maps to a token in the title in 32 different ways in the entire ODP, the five most frequent ways being to "md" (525 occurrences), "maryland" (132), "moldova" (36), "middle" (18), and "medical" (9). About 6% of the 1.3 million unique segments as recovered by Method 3 are ambiguous.

This problem is an instance of the *ambiguous abbreviation* problem which has been addressed by maximum entropy and noisy channel methods in the domains of medical abbreviations [10] and in spoken language generation for non-standard words [14]. I investigated three techniques to handle this ambiguity:

A. No expansion - pass the segments through without attempting to expand them. This method is used as a baseline.

B. Majority expansion - expand a candidate segment based on the most frequently occurring expansion found in the 1.6 million URL training corpus (e.g., all instances of "md" are mapped to the title token "md").

C. Similar abbreviation / expansion context - expand a candidate based on the similarity of its context with the possible expansions. I go into more detail about this technique below.

[10] observed that the full forms of an abbreviation are likely to appear in the same contexts as its abbreviation. Thus if an ambiguous abbreviation (e.g., "PA" in the medical domain can resolve to "Polyarteritis", "pyruvic acid", among others) occurs in the full text, a system can resolve the expansion by choosing the one with highest similarity in context to the abbreviation. To generate training data for a learner, one takes any occurrence of the candidate expansions as a positive instance of its abbreviation with the full form as the correct class.

For example, the full text might contain the token sequence "initially pyruvic acid and small amounts of atp". This is then converted to a training tuple for "PA", perhaps with a left context feature of "initially" and a right one of "and small amounts of atp". This method has an a distinct advantage over fully supervised methods that need explicitly labeled data.

I apply this method to the expansion problem by reversing the role of the full text and the abbreviation: an expansion that occurs in other URLs could also be replaced by its segment. So I make the assumption that the actual URL *www.gopusa.com/maryland* might have been written as *www.gopusa.com/md*. The URL is thus a context for $md \rightarrow maryland$.

This technique can be used to construct positive training examples for use with any machine learning method. In [10] it is paired with maximum entropy, but in this paper I choose to use the boosting algorithm, BoosTexter[12], as it performs multi-class categorization and handles text features. It also provides confidence scores with its decisions, used later in Section 6.2.

To expand an ambiguous abbreviation, a classifier must be built for it. As the training corpus contains over 89 thousand ambiguous segments, it was not possible to exhaustively build all classifiers. My objective was to quantify possible performance gain by using this method over the majority method. I selected ten frequent and ambiguous segments in the corpus for study. These segments are listed below in Table 3, along with information about their distribution.

Segment	# occur.	H(x)	Expansions
intl	220	.15	intelligence internacional international internationalization intl
asn	425	.14	abseiling ardossan asn association australasian australian austrian
ent	227	.14	ent enterprise enterpriser enterprises entertaining entertainment entomology entreprises entwicklungen
bus	201	.12	bus busaf business busing busse
tec	227	.10	teacher tec tecate tecdesign tech technical technologies technology tecnologia tecomaria tecstation tectorius
soc	274	.10	soc social sociale sociali socialism socialist societa societies society sociologia sociology socjologii
comm	254	.10	comm command commentaries commerce commercial commission commlink common communication communications communicative communist community
wm	206	.10	warmaster webmoney webmuseum weltmeisterschaft westmar westminster william williamm wm women woodmeade worldmusic
ox	380	.09	ox oxford oxfordshire oximetry oxon oxymoron
unt	220	.09	undergraduate unit united university unt

Table 3: Ten ambiguous segments selected for the pilot study. H(x) denotes entropy, defined as $-\sum p(x) * log(p(x))$, a measure of the uncertainy in the expansion of the segment.

To build each of the ten classifiers' training data, I use the ODP training data used to develop the title-based segmentation. Using this 1.6 million URL corpus, I retrieved all URLs that contained a full text expansion of the respective segment. The full text expansion was removed from each URL, and the simple segmentation (Method 1) was run on the resulting URL to get a bag of segments that were used as the context for tuples. The resulting corpus has a total of 12,348 tuples for all ten segments.

5.1 Evaluation

These tuples were presented to BoosTexter using five-fold cross validation, in which 80% of each corpus was used to train the classifiers and 20% used for testing. Each of the ten classifiers was run with 100 rounds of boosting. I report the results in Table 4 using the F1 measure, as it is a widely-used evaluation metric for multi-class categorization. F1 is defined as $\frac{2*P*R}{P+R}$, where P is precision and R is recall. I give both micro- and macro-averaged scores [16], where micro-averaging assigns equal importance to each abbreviation instance and macro-averaging assigns equal importance to each possible abbreviation expansion.

Abbrev-iation	B. Majority Expansion		C. Context + Boosting	
	macro F1	micro F1	macro F1	micro F1
wm	.20	.81	.37 (+78%)	.90 (+11%)
unt	.13	.36	.60 (+45%)	.80 (+20%)
tec	.13	.55	.35 (+74%)	.70 (+26%)
soc	.61	.31	.32 (+39%)	.59 (+88%)
ox	.25	.74	.71 (+77%)	.95 (+28%)
intl	.24	.68	.43 (+80%)	.76 (+10%)
ent	.14	.53	.42 (+95%)	.71 (+32%)
comm	.11	.64	.37 (+43%)	.77 (+21%)
bus	.36	.88	.38 (+6%)	.92 (+4%)
asn	.33	.79	.61 (+86%)	.93 (+17%)
Average	.25	.62	.45 (+82%)	.80 (+29%)

Table 4: Evaluation of Methods B and C on ambiguous segments. Parenthetical figures indicate the percentage improvement in F1 made by C over B.

The largely unsupervised context method results improves classification by a large margin. Not surprisingly, it improved the macro-average more, as the majority method fails to classify any of the $k-1$ minority categories correctly. Micro-averaged F1 improved close to 30% on average, which shows that even within the limited text in an URL, better quality disambiguation and expansion is possible.

6. PHASE III - MAPPING EXPANSIONS TO METADATA

Phase III proposes separate modules, each one to calculate a specific metadata feature. Currently only a few of the modules are implemented, although work on modules for all the metadata features shown in Figure 1 are in progress. The metadata features used in the Meurlin project are based on a study of metadata fields that are available in library cataloging summaries [6], but with differences that attempt to capture salient characteristics useful to web users. Some of the modules are very simple (e.g., media type by filename extension: PDF versus text, image, audio or animation), and others can benefit from prior work (e.g., homepage finding as done in the Ahoy! system [13]).

In the remainder of this section, I will focus on a single module in the Meurlin metadata extraction system, namely the subject/category module. This module performs the classic *text categorization* task, assigning a document into exactly one of k subjects.

To develop the categories for the text classification problem, I employ the ODP hierarchy. Ideally, one would assign a new document into one of the 477,109 hierarchically-structured categories in the ODP. In this study, I chose to limit the problem to categorizing documents of a small subset of the ODP.

I first simplified the category hierarchy by pruning it down to the top-most two levels. As such, all pages that belonged to finer grained categories were associated to their two-level ancestor (e.g., *Computers / Software / Information_Retrival / Fulltext* $\rightarrow$ *Computers / Software*). This action was taken partially to address the sparse data problems that would occur with some of the small leaf nodes as well as to simplify the problem to a coarser granularity. Of the resulting 346 categories, I randomly chose ten categories and extracted all of their 19,778 instances in the ODP.

ID	Category	# Documents (% of corpus)
122	Computers/Home_Automation	122 (>1%)
130	Computers/Desktop_Publishing	78 (>1%)
188	Health/Publications	154 (>1%)
197	Home/Rural_Living	197 (>1%)
263	Regional/Africa	12,083 (61%)
266	Regional/Central_America	2,456 (12%)
408	Sports/Fencing	626 (3%)
98999	Computers/Data_Formats	1,767 (8%)
304397	Health/Aging	134 (>1%)
321095	Health/Reproductive_Health	2,222 (11%)
	All 10 subjects	19,778 (100%)

Table 5: Random testing topics used in text classification metadata experiment. Percentages are approximate.

These ten categories are listed in Table 5. The classes are not evenly distributed, which is a characteristic common to other text categorization datasets (e.g., Reuters-21578).

I again used the BoosTexter[12] machine learner to perform the classification. To properly test the value of the different parameters in the segmentation and expansion phases as well as to bound the difficulty of the task, I constructed several different feature sets that were fed to BoosTexter[3]:

1A & 2A. Methods 1 and 2, followed by Method A - Two different methods, corresponding to the two segmentation schemes with no expansion. Method 1A represents a baseline method that establishes the lower bound for the utility of URL in the text categorization task.

2B, 3B & 4B. Methods 2, 3, 4, followed by Method B - Three segmentation schemes, paired with majority expansion.

T. Title tokens - As the majority expansion seeks to expand elements in the title, I chose to build a feature set using only the title words to upper bound the performance of expansion phase.

1A+T. Method 1A + Title tokens - A feature set that incorporates basic URL components with title tokens. Used to assess whether employing both features would improve performance.

MB. Majority Baseline - This is the decision stump of assigning all documents to the majority class *Regional/Africa*.

DW. Document words - I spidered all 19,778 URLs in the testing corpus. Each HTML document was converted into a text file via a dump of the document using the text browser, Lynx.

[3]Note that I would need to construct all 89 thousand classifier to properly include Method C, thus it was omitted from this evaluation.

This method is provided to establish the upper bound for the text categorization task.

As in the abbreviation disambiguation evaluation, each classifier was trained and tested using five-fold cross validation and 100 rounds of boosting.

6.1 Evaluation

Table 6 reports the performance of the ten feature sets, with micro- and macro-averaged F1 values. The second column gives the percentage improvement over the simple URL baseline classifier, 1A. The third column gives the normalized performance of the feature sets if MB is used as a lower bound and DW is used as an upper bound.

Method	1. F1 measure (macro/micro avg)	2. +/- % over 1A (macro/micro)	3. Normalized (macro/micro)
MB	.080/.610	-69/-10	0.00/0.00
1A	.258/.683	—/—	0.33/0.28
3B	.281/.690	8/1	0.37/0.31
2A	.362/.736	40/7	0.52/0.49
4A	.369/.738	43/8	0.54/0.50
2B	.371/.740	43/8	0.54/0.50
4B	.373/.739	44/8	0.54/0.50
T	.543/.788	110/15	0.86/0.69
1A+T	.566/.793	119/16	0.90/0.71
DW	.617/.865	139/26	1.00/1.00

Table 6: Micro- and macro-averaged F1 measure for the feature sets for the ten random subject categorization problem.

There are several observations about the results worth mentioning. The URL components by themselves do contribute sizeable percentage gain over the majority baseline. The second is that expanding segments seems to have only a slight effect on the classification process, whereas the segmentation process produces a more dramatic effect on performance. Also, the positive effect of the different segmentation schemes as evaluated informally in Section 4 directly carries over to classification performance.

With respect to the selected upper bounds, the best URL based feature set does about half as well as the title-based set and about a third as well as the document-based set. Additionally, the URL does add a measurable amount of value to the title feature set, showing that although the features overlap, the URL components offer information useful to classification that is not found in the title.

Expansion using the majority method had a small effect on classification. I investigated the cause and found two probable reasons. Although many URLs have ambiguous components, the overwhelming majority of the high frequency cases were not very ambiguous (have low entropy, e.g., *com*). Among the ambiguous cases with high entropy, the different expansions often derive from the same stem and are equally suitable resolutions. Table 7 shows this effect, in which an expansion of *wisc* to *wisconsin* makes little semantic difference.

The reader may be a bit surprised about the relatively low performance of the DW classifier. I investigated its cause and found that about 6.8% of the URLs were expired and were not retrieved, yielding documents with zero size. An additional 13.2% of the corpus had sizes less than 2 KB. An inspection of these short documents showed that many were documents with the HTML <FRAME> directive, in which classification would likely have performed better on the separate frame documents. This highlights the advantage of a URL based solution, in that the document can be classified without its existence.

Segment	# occur.	H(x)	Expansions
wisc	411	.25	[wisc wisconsin]
athletic	297	.20	[athletic athletics]
corp	194	.19	[corp corporate corporation corporations]
ltd	423	.18	[limited ltd]
adventure	232	.18	[adventure adventureland adventures]

Table 7: The five segments with highest entropy, H(x), culled from the 1,000 most frequent ambiguous segments in the 19,778 URL test corpus.

6.2 Information content based evaluation

Two further observations about the evaluation led me to perform an alternative evaluation of the feature sets. First, the learner I employed offers confidence scores, which can be used to discard scores with low confidence. A learner that abstains from a decision should not be penalized as much as one that makes an incorrect classification. Secondly, some of the categories are hierarchical, such that misclassifications leading to the same top-level category should be penalized less than ones leading to different top-level categories.

I thus also evaluated the systems using an information content approach, similar to [11]. The *information content* of a category c (employed earlier in segmentation) is inversely proportional to its probability in the taxonomy. One advantage of using this utility metric is that partial credit can be assigned to misclassified nodes that are correctly classified at a higher level in the hierarchy. This is done by assigning credit in proportional to the probability of of the ancestor level (e.g., a Health/Aging page misclassified as Health/Reproductive_Health will be assigned the IC score for Health). This metric also strikes a balance between treating categories as equally important (as in macro averaging) and treating documents as equally important (as in micro averaging). Table 8 gives the IC score and backoff scores for the 19,778-document test corpus.

Category	IC score	Backoff IC score
Computers/Home_Automation	5.08	4.59
Computers/Desktop_Publishing	5.53	4.59
Health/Publications	4.85	-
Home/Rural_Living	4.60	-
Regional/Africa	0.49	0.30
Regional/Central_America	2.08	0.30
Sports/Fencing	3.45	-
Computers/Data_Formats	2.4	-
Health/Aging	4.97	2.12
Health/Reproductive_Health	2.18	2.12

Table 8: Information content for the categories in the testing corpus.

Using this criteria, I re-evaluated the different feature sets. Here an incorrect decision subtracts the IC value of the document's class from the classifier's score. A misclassification to a category that has the same parent category as the correct class is partially correct and adds the backoff IC value to the classifier's score. Abstaining

from a decision does not affect the classifier's score.

In BoosTexter, a negative confidence score means that the document is not an instance of the class. However, in multiclass classification, the maximum value of all the confidence weights is used as the learner's decision, even when the learner has negative confidence for every class. In these cases (max confidence < 0), I regard the learner as abstaining from a decision.

New points of reference are needed to interpret system performance using the IC metric. If a learner guesses correctly (incorrectly) on every document, it would receive the maximum (-1 * maximum) possible IC credit. If a learner abstains from every decision, it receives a score of 0. Table 9 gives the IC scores for both the thresholded and non-thresholded versions of the learners on the various feature sets on the same testing set.

Method	Avg. IC Score	% Rank	Avg. # of documents abs.	corr	wrong	partial	CWR
IC worst	-5052	0	0	0	3,954	0	–
MB	-1532	34.8	0	2,413	1,056	485	4.9
1A	-281	47.2	0	2,704	846	404	6.6
3B	-142	48.5	0	2,731	823	399	6.8
Abstain	0	50.0	3,954	0	0	0	–
2A	571	55.6	0	2,914	710	330	8.8
4A	615	56.0	0	2,919	706	329	8.8
4B	637	56.3	0	2,925	700	329	8.8
2B	648	56.4	0	2,928	698	328	8.9
$1A^{T0}$	1164	61.5	1,867	1,904	118	64	29.7
$3B^{T0}$	1253	62.4	1,843	1,936	114	61	31.7
T	1715	66.9	0	3,118	537	298	10.4
$2A^{T0}$	1806	67.8	1,494	2,286	105	69	33.1
$4A^{T0}$	1826	68.0	1,477	2,299	107	70	32.8
1A+T	1829	68.1	0	3,137	524	293	10.7
$4B^{T0}$	1841	68.2	1,501	2,286	101	65	35.1
$2B^{T0}$	1847	68.2	1,473	2,305	106	69	33.4
T^{T0}	2256	72.3	384	3,046	256	268	11.3
$1A+T^{T0}$	2391	73.6	544	2,978	216	215	13.8
DW	2646	76.1	0	3,422	382	150	22.8
DW^{T0}	3151	81.1	334	3,350	140	129	25.9
IC best	5052	100	0	3,954	0	0	–

Table 9: Five-fold cross validation for the feature sets. X^{T0} denotes method X with thresholding at confidence 0. CWR = Correct to Wrong Ratio.

The IC scores validate most of the result from the standard F1 based metric, and provide perspective on the effect of thresholding. The thresholded methods dominate the IT score metrics, confirming that it prunes many incorrect answers. This is most visibly apparent in the correct to wrong ratio (CWR), as it triples in value, sometimes presenting only one incorrect judgment for every 30 answers. Thresholding also helps to close the margin between using title tokens for classification in comparison to the best URL based method – from 10% (between T and 2B) to 4% (between T^{T0} and $2B^{T0}$).

7. CONCLUSIONS

I have sketched a framework for extracting metadata from uniform resource locators. The three-phase method combines techniques from word segmentation, abbreviation expansion and applies machine learning to produce faceted metadata. Using the task of text categorization as a scenario, I have quantified the effects of different approaches in constructing machine learning features based solely on the URL. I summarize the findings below:

- Using standard the F1 measure, the performance of URL based classification is about half as effective as a title word based feature set and approximately one-third as effective as using full documents.
- The segmentation methodology greatly contributes to the overall performance of the system. The choices investigated in this paper boosts the macro-averaged F1 performance by 40% over the baseline approach. Furthermore, combining complementary segmentation schemes also improves significantly system performance.
- The expansion phase has a smaller effect. The preliminary analysis hinted that many segments often expand to words of the same stem, and which does not impact performance.

In addition to the preliminary manual evaluations and standard F1 based evaluation reported, I have used an information content evaluation that handles abstinence and assigns partial credit for misclassification in a hierarchy. The IC metric shows that thresholded version of URL-based metrics approach the performance level of using title words.

Current metadata generation and use on the Web is mostly limited to specific communities. It is also used in content blocking (such as filtering mature content[17]). It requires the cooperation of site builders, and can be an enormous expense of human labor. In contrast, the approach in Meurlin makes explicit the implicit metadata provided by the resource's URL, requiring no extra human effort.

URLs are interesting as they are often highly informative. Their small size and ubiquity on the World Wide Web make them an ideal target for study. Much current research in the Web IR community has focused on link structure, which is computationally expensive in comparison to the statistical analysis of URLs presented in this paper. While this paper shows that URLs themselves are useful for classification, in future work I plan to quantify the impact of incorporating link structure into the Meurlin system as well.

8. REFERENCES

[1] D. Bergmark. Collection synthesis. In *Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries*, Portland, Oregon, USA, 2002.

[2] S. Chakrabarti, M. van den Berg, and B. Dom. Focused crawling: a new approach to topic-specific Web resource discovery. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1623–1640, 1999.

[3] D. Connolly. Naming and addressing: URIs, URNs, ... Available from `http://www.w3.org/Addressing/`, July 2002.

[4] Y. Dai, C. S. G. Khoo, and T. E. Loh. A new statistical formula for chinese text segmentation incorporating contextual information. In *SIGIR '99*, pages 82–89, 1999.

[5] Digital Library Project. Web term document frequency and rank. Available from `http://elib.cs.berkeley.edu/docfreq/`, Last accessed January 2003.

[6] M.-Y. Kan, J. L. Klavans, and K. R. McKeown. Using the annotated bibliography as a resource for indicative summarization. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2002)*, pages 1746–1752, Las Palmas, Spain, May 2002.

[7] K. T. Lua and G. W. Gan. An application of information theory in chinese word segmentation. *Computer Processing of Chinese and Oriental Languages*, 8(1):115–124, 1994.

[8] J. Nielsen and M. Tahir. *Homepage usability: 50 websites deconstructed*. New Riders Publishing, United States of America, 2001.

[9] ODP. Open Directory Project guidelines. Information provided at: `http://dmoz.org/guidelines.html`, November 2002.

[10] S. Pakhomov. Semi-supervised maximum entropy based approach to acronym and abbreviation normalization in medical texts. In *Proceedings of the 40th Annual Meeting fo the Association for Computational Linguistics (ACL)*, pages 160–167, Philadephia, USA, July 2002.

[11] P. Resnik and D. Yarowsky. A perspective on word sense disambiguation methods and their evaluation. In *SIGLEX Workshop: Tagging Text with Lexical Semantics: What Why and How?*, pages 79–86, Washington, 1997.

[12] R. E. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 2(3):135–168, 2000.

[13] J. Shakes, M. Langheinrich, and O. Etzioni. Dynamic reference sifting: A case study in the homepage domain. In *Proceedings of the 6th World Wide Web / 29th Computer Networks Conference*, pages 1193–1204, 1997.

[14] R. Sproat, A. W. Black, S. Chen, S. Kumar, M. Ostendorf, and C. Richards. Normalization of non-standard words. *Computer Speech and Language*, 15:287–333, 2001.

[15] A. M. Turner, E. D. Liddy, and J. Bradley. Breathing life into digital archives: use of natural language processing to revitalize the grey literature of public health. In *Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries*, page 411. ACM Press, 2002.

[16] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979.

[17] World Wide Web Consortium. Platform for Internet Content Selection (PICS). Available from `http://www.w3.org/pics/`, Last accessed January 2003.