A general learning method for automatic title extraction from HTML pages

Sahar Changuel, Nicolas Labroche and Bernadette Bouchon-Meunier

Laboratoire d'Informatique de Paris 6 (LIP6)
DAPA, LIP6
104, Avenue du Président Kennedy, 75016, Paris, France
{Sahar.Changuel, Nicolas.Labroche, Bernadette.Bouchon-Meunier}@lip6.fr

Abstract. This paper addresses the problem of automatically learning the title metadata from HTML documents. The objective is to help indexing Web resources that are poorly annotated. Other works proposed similar objectives, but they considered only titles in text format. In this paper we propose a general learning schema that allows learning textual titles based on style information and image format titles based on image properties.
We construct features from automatically annotated pages harvested from the Web; this paper details the corpus creation method as well as the information extraction techniques.
Based on these features, learning algorithms, such as Decision Trees and Random Forest algorithms are applied achieving good results despite the heterogeneity of our corpus, we also show that combining both methods can induce better performance.

1 Introduction

With the rapid increase of information spreading on the Web, locating the relevant resources is becoming more and more difficult. One approach to make the Web more understandable to machines is the Semantic Web[1], where resources are enriched with descriptive information called metadata.
Metadata is commonly known as a kind of structure data about data that can describe the content, semantics and services of data [1], playing a central role in supporting resources description and discovery. Basic metadata about a document are: its title, its author, its publisher, its date of publication, its keywords and its description [2].
Although manual annotations are considered as the main source of information for the Semantic Web, the majority of existing HTML pages are still poorly equipped with any kind of metadata. Hence automatic metadata extraction is an attractive alternative for building the Semantic Web.
The three main existing methods to generate metadata automatically are [3]:

- Deriving metadata: creating metadata based on system properties.

[1] `http://www.w3.org/2001/sw/`

- Harvesting metadata: gathering existing metadata, ex: META tags found in the header source code of an HTML resource.
- Extracting metadata: pulling metadata from resource content; metadata extraction occurs when an algorithm automatically learns metadata from a resource's content. Automatic extraction may employ sophisticated indexing and classification algorithms to improve the metadata quality.

In this paper we focus on title extraction from HTML documents as part of a more global application of automatic metadata extraction from learning resources.
Title is a field that we can find in most metadata standards schemas: in Dublin Core[2], MARC[3], MODS[4], LOM-fr[5], EAD[6], etc.
It is important to have the correct value of the title since it provides information on what a document is about providing scalability and usability for the resource.

We propose an automatic title extraction method based on supervised machine learning technique such as Decision Trees and Random Forests methods. A well known drawback of the supervised machine learning method is the manual annotation of the input data set. In this paper, we reuse the knowledge embedded in the header of the HTML source code in order to obtain labeled training data for title extraction with limited human effort.

Html pages can have their titles in text or in image format. In figure 1, the title of the page 'a)' is contained in an image , whereas the title of the page 'b)' is in text format.

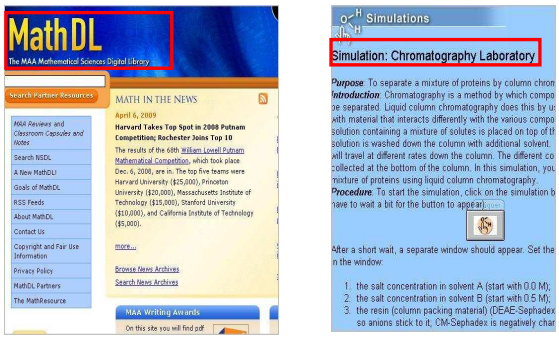

Fig. 1. Example of HTML pages titles

[2] Dublin Core Metadata Initiative, `http://dublincore.org/`
[3] Machine Readable Cataloging, `http://www.loc.gov/standards/`
[4] Metadata Object Description Schema, `http://www.loc.gov/standards/mods/mods-schemas.html`
[5] Lom-fr, `http://www.lom-fr.fr/`
[6] Encoded Archival Description, `http://www.loc.gov/ead/`

To extract text format titles, style information such as font size, position, and font weight are used as features. We also proposed a new method to extract image format titles, which is a more challenging task since fewer information can be extracted from images, the 'alt' attribute was used to get the text content of each image in the labeling phase.

The rest of the paper is structured as follows: In section 2, we introduce related works on automatic metadata extraction. In section 3, we explain the corpus creation and the feature extraction for the two methods of title extraction (text and image titles). Section 4 describes the classification algorithms we used and presents the obtained results . We make concluding remarks and highlight future research directions in section 5.

2 Related work

Several methods have been used for automatic metadata extraction, most of them use one of the two approaches: harvesting or Machine Learning.

To extract metadata, and especially the title, from HTML documents, systems generally harvest information from the header part of the HTML source code, this include Klarity and DC.dot [4] applications which generate metadata automatically from the author-supplied Meta tags in HTML documents. We also cite the MetaCombine Project [5] which uses an intuitive approach to generate some basic metadata tags; it checks for $< title >$ tag in page, if present and nonempty, stores title in metadata, otherwise it considers the first 64 characters of the plain text as the title.
A current limitation of the harvesting method is that the metadata elements are not always populated by resource creators or other persons. In our experiments, we found that 26.36% out of 2367 HTML pages contain correct values of the Meta $< title >$ fields and that 51.42% of them have their titles in the first line text.

In addition to harvesting existing information, Machine learning technologies have been used for automatic metadata extraction, authors in [6] proposed a method to conduct metadata extraction from header part of scientific research papers. They formalized the problem as that of classification and employed Support Vector Machines as a classifier using mainly linguistic features in the model.

Fewer researchers were interested in extracting title separately as a metadata field, the paper in [7] describes a content-based and domain-independent method using Support Vector Machine for title extraction from Chinese scientific papers. They use bigram-specific features and sentence-specific features. Whereas, in [8], authors proposed a machine learning approach to title extraction considering Microsoft Office as a case of study and using formatting information such as font size as features in their models.

In [9] authors proposed a machine learning method for title extraction from HTML pages based on format information. Our method is different in that we use a domain independent corpus composed of HTML pages labeled automatically without human effort. Moreover, to construct our feature vectors, we used

fewer attributes by selecting, experimentally, the most informative ones, and we applied Decision Trees and Random Forests algorithms on these features. Furthermore, we have proposed a new method for image format title extraction.

Experimental results indicate that both methods can generate good results (in term of F-measure) and performs well when applied to a different data sets which indicates that our method is domain independent. We also show that combining both methods can improve the results and can give higher performance to title extraction.

3 Pre-processing

Preparing input for a data mining investigation usually consumes the bulk of the effort invested in the entire data mining process [10].
To avoid human annotation, we reuse the title tag embedded in the header of the HTML source code in order to obtain labeled training data with limited manual effort. In this section we explain our corpus creation methodology.

3.1 Text format title extraction

3.1.1 Corpus creation

To create our corpus, the Meta $< title >$ tag (M_tag) is used from the HTML source code. Because this tag is generally absent or badly annotated, we need access to a large collection of HTML pages, which will be filtered depending on the title labeling correctness.
Since the web is one of the most important knowledge repository, we decided to use it to construct our data set.
Pages are harvested from the Web as follows:

- A list of queries are submitted to a search engine using the Argos library[7].
- The system returns a list of web pages URLs.
- The pages having correct M_tag values are selected.

• Queries submission
Within the context of our global application of automatic metadata extraction from learning resources, we are especially interested in extracting information from the education domain, thus, the words chosen in the queries belong to the education lexicon. Ex:

- english+courses+student
- history+geography+courses
- chemistry+courses

Queries are also constructed from French words so as to construct a mixed corpus, and to have a language independent methodology. Ex:

- anglais+exercices+licence

[7] `https://argos.dev.java.net/`

- chimie+cours
- physique+exercices+licence

The choice of the educational domain is in no way a restriction to our extraction approach, our method is conceived to be domain independent.
Each query is submitted to the Web search engine and we get back a list of URLs as a result.

• HTML page parsing
For each result page, the corresponding *M_tag* value is extracted. In order to analyze an HTML page for content extraction, its is passed first through an open source HTML syntax checker, Jtidy[8], which corrects the markup, transforming the ill-formed HTML document to a well-formed XML one.
The resulting document is then parsed using the Cobra toolkit[9], which creates its Document Object Model Tree[10] (DOM tree) representation.
We consider the $< body >$ HTML element as the root node of the tree. Our content extractor navigates the DOM tree and get the text content from the leaf nodes.
A set of filters is used to ignore some tags or some specific attributes within tags like links, scripts, drop down menu, and many other elements from the page. Eliminating these elements avoids us reading useless nodes.
For each page, we consider only the 20 first text nodes from the DOM tree, assuming that the title is in the top part of the page.
Each text is then compared to this value using a similarity measure.

• Similarity measure
To compare a text *"txt"* with the *M-tag* value (*m-txt*) of a page, we calculate the proportion of words from *txt* which are in *m-txt*. We suppose that *txt* can be a title if this proportion is greater than a threshold that has been experimentally set to 0.7.
More formally, suppose T the set of words in *txt* and M the set of words in *m-txt*. *Txt* is considered similar to *m-txt* if the following equation is verified:

$$\frac{\sum_{t \in T} s(t, M)}{|T|} > 0.7 \quad (1)$$

Where $s(t, M) = 1$ if $\exists\ m \in M / t = m$, and 0 otherwise.
Unlike [9] our method doesn't use the edit distance as a measure of similarity since, in addition to the title of the page, *m-txt* can contain other words describing the page, this can penalize the score of the real title if it exists.
For an HTML page, the first text verifying the previous condition is considered as the title of the page, hence, its URL and the corresponding M-tag value are saved in our corpus.

[8] `http://jtidy.sourceforge.net/`
[9] `http://lobobrowser.org/cobra.jsp`
[10] `http://www.w3.org/`

This treatment is repeated for each page result of each query to obtain finally 624 correctly annotated pages from 2367 pages. This corpus is called the *net-txt* corpus. The different steps of the corpus creation are summarized in figure 2.

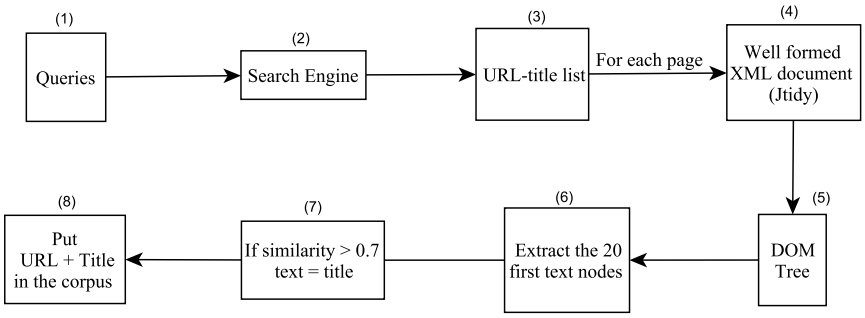

Fig. 2. Textual title corpus creation phases

Another collection of documents is also used to construct a test data set in order to evaluate the classifier. The Eureka[11] collection is used for this purpose. It is an online educational portal offering resources with well annotated metadata in XML format respecting the LOM (Learning Object Metadata)[12] schema. The URLs and the titles are extracted from these XML documents using XPATH queries, then the labeling process is continued by following the schema of the figure 2. 424 correctly annotated pages are obtained out of 1633, we call this corpus the *Eureka-txt* corpus.

3.1.2 Features extraction

In metadata extraction systems, title is generally extracted from the following sections of the HTML document:

- The M-tag value.
- The first line of the document.
- H1 and H2 tags.
- H1 tags.

As mentioned previously, few pages contain correct M-tag values, in fact only 26.36% of the pages resulting from the different queries contain the correct values, thus considering only the M-tag information for title extraction is not reliable.

[11] `http://eureka.ntic.org/`
[12] `http://www.lom-fr.fr/`

The other methods are tested on our corpus and the results are summarized in table 1.
The columns list the title extraction method, the percentage of titles extracted with each method on the *Net-txt* corpus, on the *Eureka-txt* corpus, and on both corpora.
The table shows that none of the methods listed is efficient for extracting the titles from HTML pages, other properties should be considered.

Table 1. Performances of different methods for title extraction

	Net-txt	**Eureka-txt**	**Both corpora**
First line title	44.36%	58.48%	51.42%
H1-H2 tags	49.28%	35.95%	42.61%
H1 tags	36.64%	30.87%	33.75%

Our method is based on style text properties to construct the features. The Cobra java toolkit[13] is used to extract these information; the toolkit is a CSS-aware HTML DOM parser and allows to get the style of each text i.e. the size, the alignment, the colour, etc.
36 features are constructed based on the following information:

- Current node information, such as letters capitalization, font size, font weight (bold, strong, emphasized), colour, alignment, H1 tag, etc.
- Format change with the previous and the following text: family font change, size change, alignment change ...
- The proportion of the format compared to the rest of the page: size, family font, colour, etc.

For each text node a feature vector is constructed. The last element of this vector is the class, i.e. whether the text is the title of the document or not, based on its similarity with the annotated title.
More than one text can have a similarity measure greater that 0.7 in the same document, in that case, the text with the biggest font size is considered as the title of the page.
In order to avoid a significant gap between the number of instances assigned to the class "title" and those assigned to the class "non title", the 20 first instances are stored for each page.
8408 examples are obtained from the *net-txt* corpus, we call them *net-txt* attributes, and 4858 examples are acquired from the Eureka corpus, they are called *Eureka-txt* attributes.

[13] `http://lobobrowser.org/cobra.jsp`

3.2 Image format title extraction

As mentioned before, an HTML page title can either be a text or an image. Extracting image format titles requires acquiring their text contents. A well known method for translating printed text into machine-editable text is the Optical Character Recognition method usually abbreviated to OCR, but this is out of the scope of our work.
The alternative solution proposed in this paper focuses on images with filled 'alt' attributes. The alt attribute is an essential part of the Web accessibility, providing a text equivalent for every non-text element[14].
For an image containing text in some specific font style, that text is used as the value of the alt attribute. An example of alt attribute given by the HTML 2.0[15] is: <IMG SRC="triangle.xbm" ALT="Warning: Be sure to read these instructions">.
In this paper, the alt attribute is used in order to extract image title from HTML pages. The issue is that this attribute is rarely filled, which makes the task a challenging problem.

3.2.1 Corpus creation

We need a set of HTML pages having images titles with filled alt attributes. To create our corpus, the previous method based on querying the Web is adopted, but in spite of getting text nodes from each page, we get the alt attribute values from image nodes. Each one is compared with the M-tag value using the similarity measure defined in equation 1.
Few pages satisfying the requirement are obtained (148 pages out of 7471 pages resulting from the queries), the reason is that people seldom specify the alt attribute when adding images to HTML pages. We call this corpus the *net-img* corpus.
The same method is adopted to get 51 pages from the Eureka corpus, we call this corpus the *Eureka-img* corpus.

3.2.2 Features extraction

For image format titles, the style properties of the text contained in images can't be obtained, thus, other information should be used for feature construction.
In our method, we extract spatial features based on the following properties:

- Image alignment.
- Image height and width.
- Image surface.
- Image position in the page.
- H1 tag information.

13 features representing each image are constructed. The last element is the class of the feature: whether the image is the title or not; the alt text is compared

[14] `http://www.w3.org/TR/WAIWEBCONTENT/`
[15] `http://www.w3.org/MarkUp/htmlspec/htmlspec_toc.html`

with the page title using the equation 1.
Feature extraction is done on both corpus: the *net-img* and the *Eureka-img* corpus, to get 743 instances from the first and 214 from the second.

4 Algorithms and evaluations

4.1 Algorithms

A supervised learning technique is used to extract titles from HTML pages. Let $\{(x_1, y_1)...(x_n,y_n)\}$ be a two-class training data set, with x_i a training feature vector and their labels y_i (1 for the class 'title' and -1 for the class 'non title'). Experiments are conducted using two classifiers, Decision Tree and Random Forest, the results are compared for each corpus.

• Decision Tree algorithm
We use the C4.5 algorithm implemented in Weka[16], since in addition to nominal attributes it can deal with numeric attributes and with noisy data.

Decision Tree algorithm works with a top down approach, seeking at each stage an attribute to split on that best separates the classes, then recursively processing the subproblems that result from the split.
The information measure used as a basis for evaluating different splits is the entropy which characterizes the (im)purity of each arbitrary collection of instances.

A set of rules is generated, one rule is generated for each leaf. The antecedent of the rule includes a condition for every node on the path from the root to that leaf, and the consequent of the rule is the class assigned by the leaf.
Rules derived from trees are pruned to remove redundant test, the C4.5 algorithm adopts a strategy of post-pruning by building the complete tree and pruning it afterwards [10]. In our experiments, the confidence factor used for pruning is 25%.

• Random Forest algorithm
Random Forest algorithm was developed by Leo Breiman [11], to operate quickly over large datasets. More importantly, it can be diverse by using random samples to build each tree in the forest and combining the classifiers predictions by voting.
In the Random Tree method, a tree is constructed as follow [12]:

1. Instances are randomly sampled, with replacement, from the original data set to create a new one of the same size to be used for tree construction (inBag).
2. Choose a random number of attributes k from the inBag data and select the one with the most information gain to comprise each node.
3. Continue to work down the tree until no more nodes can be created due to information loss.
4. Combine the trees by having them vote on each test instance. If one class receives more votes than any other, it is taken as the correct one.

[16] `http://www.cs.waikato.ac.nz/ml/weka/`

We note N the total number of attributes. If not specified, k is equal to the first integer less than $log_2(N) + 1$.
We use the Random Forest algorithm implemented in Weka which is based on the C4.5 Decision Tree algorithm, 10 is chosen as the number of Decision Tree classifiers.
Using Random Forest algorithm should make the outcome of classification more stable by taking profit of the complementarily of the different trees.

4.2 Evaluations

This section provides empirical results to evaluate the two methods of title extraction described previously.
Performance is evaluated by precision and recall described as follows:

- A : The number of items correctly labeled as belonging to the class 'title'.
- B : The number of items incorrectly labeled as belonging to the class 'title'.
- C : The number of items not labeled as belonging to the class 'title' but which should have been.

Precision measures the number of correctly identified items as a percentage of the number of items identified: $Precision = \frac{A}{A+B}$.
Recall measures the number of correctly identified items as a percentage of the total number of correct items: $Recall = \frac{A}{A+C}$.
Precision can be seen as a measure of exactness or fidelity, whereas Recall is a measure of completeness.
Both are combined into a single measure: Fl-measure metric which is the weighted harmonic mean of precision and recall:

$$F1 - measure = \frac{2 * Precision * Recall}{Precision + Recall}$$

F-measure exhibits the desirable properties of being highest when both recall and precision are high .

4.2.1 The text title extraction method

For text title extraction, both Decision Tree and Random Forest classifiers are used in our experiments.
Initially, a 10-fold cross validation is conducted on the *net-txt* attributes and on the *Eureka-txt* attributes separately.
Suppose k the number of attributes to be used in random selection for the Random Forest classifier: $k = log_2(36) + 1 = 6$. The classifier is first tested with the default value of k, then tests are made with different values of k, we mention the best results obtained. Table 2 summarizes the results.

Table 2 shows that the performance of title extraction on the *Eureka-txt* corpus is better than that on the *net-txt* corpus in term of precision, recall and F-measure. This can be explained by the fact that pages of the *Eureka-txt* corpus

Table 2. Cross validation results of the text title extraction method

Corpus	Methods	Precision	Recall	F1-Measure
Net-txt	Decision tree	0.808	0.619	0.701
	Random Forest k=6	0.828	0.72	0.770
	Random Forest k=13	0.847	0.729	0.784
Eureka-txt	Decision tree	0.890	0.840	0.864
	Random Forest k=6	0.896	0.873	0.884
	Random Forest k=13	0.903	0.882	0.893

belong to the same collection and share common patterns, whereas, the *net-txt* corpus contains domain independent pages belonging to different collections and sharing fewer stylistic patterns.

Results performed by the Random Forest models are better than those generated by Decision Tree model. With the Random Forest algorithm the trees correlation provides a higher performance and predictions become more reliable as more votes are taken into account.

The default number of features does not give the best results when using the Random Forest classifier, increasing k improves the performance of the model even if it leads to a higher correlation between the trees.

We remark that our method is language independent since the *net-txt* corpus contains both French and English pages.

To test the domain adaptation of our method, the model is first applied with the *net-txt* corpus data as input, and the *Eureka-txt* data set for test, then we invert the training and the test data sets. Results are summarized in table 3.

Table 3. Domain adaptation of the text title extraction method

Corpus	Methods	Precision	Recall	F1-Measure
Net-txt/Eureka-txt	Decision Tree	0.834	0.748	0.789
	Random Forest k=6	0.830	0.748	0.787
	Random Forest k=13	0.810	0.785	0.798
Eureka-txt/Net-txt	Decision Tree	0.722	0.554	0.627
	Random Forest k=6	0.661	0.575	0.615
	Random Forest k=13	0.696	0.601	0.645

The results of the *Net-txt/Eureka-txt* experiment show that our extraction model is domain independent, the *net-txt* corpus contains heterogeneous pages from different domains making the model sufficiently generalized.
When swapping the two data sets the result decreases in term of precision, recall and F-measure, the reason of that is that pages from the *Eureka-txt* corpus share some common patterns.

4.2.2 The image title extraction method

10-fold cross validations are conducted, using Decision Tree and Random Forest classifiers. With the former, the default number of attributes is used in the random selection i.e. $k = int(log_2(13) + 1) = 4$.
Table 4 shows the results obtained when using the 743 attributes generated from the *net-img* corpus.

Table 4. Cross validation results of the image title extraction method

Method	Precision	Recall	F-Measure
Decision Tree	0.865	0.779	0.820
Random Forest	0.870	0.820	0.844

Both classifiers give good results and make an effective use of the image information for title extraction.
Random Forest classifier is also evaluated using the 214 attributes from Eureka pages as a test set and the *net-img* corpus attributes for training. Results are summarized in table 5. The Results obtained with the Random Forest classifier are slightly better than those we get with the Decision Tree algorithm when using different data sets for training and for testing. This reproves that our method is domain independent.

Table 5. Domain adaptation of the image title extraction method

Method	Precision	Recall	F-Measure
Decision Tree	0.705	0.705	0.705
Random Forest	0.833	0.795	0.814

The results above show that our method can still achieve a relatively high performance when titles are in image format. The technique employed in this paper, though simple, is quite effective.

4.2.3 Text versus image format title extraction

Further investigation is done to analyze the relation between the text title extraction method and the image title extraction one, our aim is to test to what extend each method improves the other.
For this purpose, to test both methods on the same corpus, the corpus we need should contain pages with both formats of titles: image and text. This corpus is obtained by filtering the *net-img* and the *Eureka-img* corpus.
Our aim is to apply the text title extraction method to this corpus on the one

hand, and the image title extraction method on the other hand. Hence, image attributes as well as text attributes are extracted from these pages, we call the text attributes *txt-img* attributes.
Experiments are conducted on these features using Decision Tree classifier. We did not use Random Forest in this experiment since our data set is small.
Firstly, the model trained with the *net-txt* attributes is applied to the *txt-img* attributes; we call this experiment *Txt-Exp*.
Second, since our data set is not large, a 10-fold cross validation is conducted with the image attributes, this experiment is called *Img-Exp*.
The results obtained are summarized in table 6.

Table 6. Performance of text vs. image title extraction methods

Experiment	Precision	Recall	F-Measure
Txt-Exp	0.742	0.529	0.617
Img-Exp	0.833	0.68	0.749

Txt-Exp gives less performing results than *Img-Exp*, this is due to the fact that the pages of the data set have essentially image titles and the annotated text titles are sometimes ambiguous and can be confused with the rest of the text.

Fig. 3. Text versus image format title extraction

When further analyzing the results, we notice that the text method can sometimes extract the title of a page whereas the image method could not extract

Fig. 4. Text versus image format title extraction

the image title and vice versa. This leads us to suppose that combining both methods can significantly improve the results. Figures 3[17] and 4[18] show this more clearly.

In figures 3, the image title was not found by the image classifier whereas the text title was extracted by the text method classification.
In figures 4, we notice the opposite, the image title was found and the text title was not.

Combining both methods can lead to a higher performance; to test this, we need a bigger corpus, and a large number of HTML pages with images having filled alt attributes, however, as mentioned before, this attribute is seldom used by HTML pages creators and there is still a lack of awareness of its importance especially for accessibility purpose. An alternative solution would be to use an optical character recognition method.

5 Conclusions

Under an application of automatic metadata extraction, we notice that title is a field which is present in all metadata schemas, and which is widely used in search engine applications.
This paper describes two techniques for HTML pages title extraction based on machine learning methods. The first approach extracts text format title based on style properties, and the second extracts image format title using the alt attributes of $< img >$ tags.
A method for corpus creation was proposed. It is based on extracting pages by

[17] `http://www.clg-armand.acaixmarseille.fr/spip/`
[18] `http://www.polarfle.com/`

querying the Web, it guarantees that the obtained pages have diverse patterns, different languages and deal with distinct subjects.
This paper shows that, on two data sets, our methods perform well in most cases, it shows also that combining text and image extraction methods can lead to better results, we suggested using OCR techniques to expand the data set of image titled pages .
Future directions include discovering other fields of metadata from HTML pages so as to enrich resources and to make them more accessible.

References

1. Liu, L., He, G., Shi, X., Song, H.: Metadata extraction based on mutual information in digital libraries. In: Information Technologies and Applications in Education, 2007. ISITAE '07. First IEEE International Symposium on. (2007)
2. P.P, N.: Metadata: Automatic generation and extraction. 7th MANLIBNET ANNUAL NATIONAL CONVENTION on Digital Libraries in Knowledge Management: Opportunities for Management Libraries, at Indian Institute of Management Kozhikode (2005)
3. Greenberg, J., Spurgin, K., Crystal, A.: Functionalities for automatic metadata generation applications: a survey of metadata experts' opinions. Int. J. Metadata Semant. Ontologies **1** (2006) 3–20
4. Greenberg, J.: Metadata extraction and harvesting: A comparison of two automatic metadata generation applications. Journal of Internet Cataloging **6** (2004) 59–82
5. Krowne, A., Skinner, K., Halbert, M., Ingram, S., Gadi, U., Pathak, S.: Metacombine project interim report. Technical report, Emory University (2006)
6. Han, H., Giles, C.L., Manavoglu, E., Zha, H., Zhang, Z., Fox, E.A.: Automatic document metadata extraction using support vector machines. In: Digital Libraries, 2003. Proceedings. 2003 Joint Conference on. (2003) 37–48
7. Zhang, Z., Sun, M., Liu, S., eds.: Automatic content based title extraction for Chinese documents using support vector machine, Proceedings of 2005 IEEE International Conference on (2005)
8. Hu, Y., Li, H., Cao, Y., Teng, L., Meyerzon, D., Zheng, Q.: Automatic extraction of titles from general documents using machine learning. Inf. Process. Manage. **42** (2006) 1276–1293
9. Hu, Y., Xin, G., Song, R., Hu, G., Shi, S., Cao, Y., Li, H.: Title extraction from bodies of html documents and its application to web page retrieval. In: SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM (2005) 250–257
10. Ian H. Witten, E.F.: Data Mining: Practical Machine Learning Tools and Techniques (Second Edition). Diane Cerra (2005)
11. Breiman, L.: Random forests. In: Machine Learning. (2001)
12. Pater, N.: Enhancing random forest implementation in weka. In: Learning Conference Paper for ECE591Q. (2005)