

# AIR: a Semi-Automatic System for Archiving Institutional Repositories

Natalia Ponomareva<sup>1</sup>, Jose M. Gomez<sup>2</sup>, and Viktor Pekar<sup>3</sup>

<sup>1</sup> University of Wolverhampton, UK,  
nata.ponomareva@wlv.ac.uk

<sup>2</sup> University of Alicante, Spain,  
jmgomez@ua.es

<sup>3</sup> Oxford University Press,  
viktor.pekar@oup.com

**Abstract.** Manual population of institutional repositories with citation data is an extremely time- and resource-consuming process. These costs act as a bottleneck on the fast growth and update of large repositories. This paper aims to describe the AIR system developed at the university of Wolverhampton to address this problem. The system implements a semi-automatic approach for archiving institutional repositories: firstly, it automatically discovers and extracts bibliographical data from the university web site, and, secondly, it interacts with users, authors or librarians, who verify and correct extracted data. The system is integrated with the Wolverhampton Intellectual Repository and E-theses (WIRE), which was designed on the basis of standard software adopted by many UK universities. In this paper we demonstrate that the system can considerably increase the intake of new publication data into an institutional repository without any compromise to its quality.

## 1 Introduction

The continued development of OAI-PMH (Open Archive Initiative Protocol for Metadata Harvesting) compliant institutional repositories is of vital importance to education and research institutions and to the economy in general<sup>4</sup>. Many institutions that have implemented repositories have encountered cultural barriers to embedding repositories and challenges surrounding deposit. The university of Wolverhampton has successfully established WIRE - an open access institutional repository for research output<sup>5</sup>. Open Repository is a service from BioMed Central to build, launch, host and maintain institutional repositories for organisations. Built upon the latest DSpace repository software<sup>6</sup> the service has been designed to be flexible and cost-effective.

The WIRE repository was launched two years ago and it is already populated with over 1500 articles and theses. However, we are acutely aware both from our

<sup>4</sup> [http://www.jisc.ac.uk/news/stories/2007/06/news\\_repos.aspx](http://www.jisc.ac.uk/news/stories/2007/06/news_repos.aspx)

<sup>5</sup> <http://wlv.openrepository.com/wlv/>

<sup>6</sup> <http://www.dspace.org/>

own experience and through talking to WIRE librarians that we are still far away from comprehensive archiving (or as close as we can get to this while remaining within the constraints of copyright, licensing and embargo restrictions). Common difficulties facing many repositories include the identification and capture of their institution's research output, time spent providing mediated deposition services or resistance to, and/or lack of engagement with, self-archiving and copyright/licence clearing of material for open access.

The AIR (Automatic Archiving for an Institutional Repository) system was developed to address the problems mentioned above and was implemented based on the essential principals of software development: platform independence, reliability, portability, easy customisation, scalability, etc. It employs an automatic approach to identify publications and extract bibliographical metadata from them in order to assure completeness of the research data produced by the university staff. The quality of the bibliographical metadata uploaded to WIRE is ensured by authors and WIRE librarians who can access, verify and, if necessary, correct the extracted information with the help of a user friendly interface.

The paper is organised as follows. In Section 2 an overview of related works is provided. General system architecture and its main characteristics are reported in Section 3. Section 4 is devoted to the detailed description of the AIR core module, whose evaluation is given in Section 5. Finally, in Section 6 we draw conclusions and discuss the future work.

## 2 Related works

In response to time- and resource constraints of manual population of institutional repositories, a number of research projects have been recently carried out. There can be distinguished two main approaches to create and populate digital repositories. One of them exploits the principal of collaborative sharing bibliographical metadata. For example, CiteULike<sup>7</sup> is based on social bookmarking: researchers create their own databases and share them with other users. This system has a facility of importing articles from different online resources with automatic detection of bibliographical metadata. A similar principal is used by Bibster<sup>8</sup> - an ontology based Peer-to-Peer system for exchanging bibliographical data among researchers. It uses two different ontologies: one contains some generic aspects of bibliographical metadata and the other describes specific categories for the Computer Science domain. The use of these ontologies enables automatic extraction of metadata from bibliographical entries and provides an effective search for required publications ranking the results according to their semantic similarity with a query [1].

Another approach that was adopted by many existing systems exploits an automatic search of bibliographical metadata on the Web. For example, Symplectic Publications Management System<sup>9</sup> (PMS) automatically harvests

<sup>7</sup> <http://www.citeulike.org/>

<sup>8</sup> <http://bibster.semanticweb.org/>

<sup>9</sup> <http://www.symplectic.co.uk/products/publications.html>

journal citation data from some online sources such as ISI Proceedings, PubMed and Web of Science. Nonetheless, most of these systems can only make use of data that has already been encoded in the format with which they have been adapted to deal. As a result, the coverage of such systems is greatly limited. Symplectic PMS, for example, processes only journal papers of specified publishers, missing out such publications as conference proceedings, books and book chapters.

To our knowledge, the only systems that actually extract bibliographical data from the unrestricted Web are CiteSeer<sup>10</sup> and Google Scholar<sup>11</sup>.

### 3 System architecture

General architecture of the AIR system is presented in Fig. 1. The process of extracting bibliographical metadata can be described as follows: first, a crawler populates the database with all web pages belonging to the university domain. The extracted web pages go to the AIR core module, which classifies pages into relevant (containing bibliographical entries) and irrelevant ones, then finds references and maps their parts with Dublin Core Metadata tags. We propose a semi-automatic approach that ensures the reliability of information transferring to WIRE: automatically extracted data must be verified by users in order to be uploaded to WIRE. The verification process is carried out through user-friendly web interfaces. The final step of the system consists in transferring the fully revised data to WIRE, which was implemented using SWORD<sup>12</sup> protocol.

#### 3.1 Principal modules

In order to crawl web pages belonging to the university domain we used Nutch software<sup>13</sup>: an open source Java implementation of a search engine. It provides all necessary tools to run a search engine satisfying some user requirements. Nutch is built on top of Lucene, which is an API for text indexing and searching, and it is naturally divided into two pieces: the crawler and the searcher. The crawler fetches pages and turns them into an inverted index, which the searcher uses to answer users' search queries. The crawler can fetch billions of web pages quickly and efficiently, using a distributed filesystem via Hadoop<sup>14</sup>. We employed Nutch mostly because it claims to be a transparent and stable tool with an open source implementation and it is widely used by many researchers.

The AIR core module (Fig. 2) is the principal module of the AIR system as it accomplishes an automatic search for publications in the university web site. Due to the complex structure of the AIR core module it will be described in detail in a separate section (see Section 4).

---

<sup>10</sup> <http://citeseer.ist.psu.edu/>

<sup>11</sup> <http://scholar.google.co.uk/>

<sup>12</sup> [http://www.ukoln.ac.uk/repositories/digirep/index/SWORD\\_guide](http://www.ukoln.ac.uk/repositories/digirep/index/SWORD_guide)

<sup>13</sup> <http://lucene.apache.org/nutch/>

<sup>14</sup> <http://hadoop.apache.org/core/>

**Fig. 1.** General system architecture

**Fig. 2.** The AIR core module design

The Web interfaces were developed in order to provide user interaction with the AIR system. The author of each publication needs to verify the automatically extracted data, post-edit it if necessary, and approve the deposit of the data into the institutional repository. Once authors log into the system (after being notified by email that their new publications are ready to be deposited into the repository), bibliographical data from the database is displayed in the

web interface form that authors would normally fill in manually. For authors' convenience, they can see the actual document from which the data has been extracted. Authors can change the text in the form and then submit it for further approval by WIRE librarians. The underlying idea is that instead of typing in whole entries in the web form, authors only have to edit the automatically extracted data that is already inserted into the appropriate fields of the form, in this way speeding up the process.

The web interface via which WIRE librarians verify the data submitted by authors has additional features that can facilitate the process of verifying and uploading information to WIRE. Firstly, it provides a facility that automatically queries the SHERPA/Romeo database<sup>15</sup> to determine the copyright status of publications. Secondly, it carries out an automatic web search of publication's identifiers (ISSN, ISBN, etc.) using Yahoo APIs. The search is based on four elements: identifier format, publication's title, closeness of identifiers (that satisfied the format) to the title and frequency of identifier appearance in retrieved pages.

The possibility of transferring validated data to WIRE is delivered by SWORD facility which requires the bibliographical data to be encoded according to Dublin Core Metadata format and written to an XML-file of a special structure dictated by SWORD. An XML-file is created separately for every publication, then it is compressed to a ZIP file and sent to WIRE.

### 3.2 System characteristics

The AIR system was developed and implemented to satisfy the following requirements:

*1.Platform independence.* The system was implemented in such a manner that it can be easily installed and run under any platform with minimum effort from the user.

*2.Reliability.* The reliability of the AIR system was ensured through careful definition of its requirements, separating the processes of software design and implementation and careful testing.

*3.Easy customisation.* The AIR system was designed in such a way that all its parameters are set in external XML-files that can be easily changed without any code modification.

*4.Strong object orientation and code clearness.* This is an important feature if the code has to be modified and adjusted for some requirements. Object orientation allows the code to be flexible and simplifies modifications and additions.

*5.Portability.* Portability is a very important issue as it refers to an easy adaptation of the AIR system to other institutional repositories. This characteristic has to be taken into account due to the same standards used for implementation of institutional repositories in the UK. In order to archive portability the data extraction components of the system were not fine-tuned to

---

<sup>15</sup> <http://www.sherpa.ac.uk/>

web page layouts and formats established at the university of Wolverhampton. Moreover, machine learning algorithms, we employed in order to train the extraction components, also assure easy system adjustment to other formats or even domains.

*6. Scalability.* Experiments with the system showed that it is scalable and can easily cope with large amounts of data (i.e. web pages and publications). Currently the only restrictions identified are due to the Nutch engine used for crawling web pages and MySQL database where all publications are stored. Both programs are mature pieces of software that can be employed to process large quantities of data.

*7. Security issues.* The system must be secure and prevent the transfer of invalid or junk information to the WIRE. Therefore, librarians, as the only users who have the right to upload data to WIRE, are provided with logins and passwords. The passwords are coded and securely stored on the server. For the sake of security librarians are given different access rights to the system. Only librarian-administrators can create new users and allow them to enter the system.

*8. Extracting information types corresponding to Dublin Core Metadata.* The system aims to extract the following types of information: Author name, Title, Year of Issue, Identifier, Publisher, Citation (which includes Conference/Journal/Book Name, Volume, Pages, Venue, etc.). It should be highlighted that the system can be easily adapted to other types of information due to its machine-learning based implementation.

## 4 AIR core module

The AIR core module consists of 5 submodules:

1. Page classifier that extracts web pages containing bibliography.
2. Record classifier that selects bibliographical entries from all document records.
3. Information Extraction (IE) module which aims to map bibliographical data with Dublin Core Metadata fields, i.e. title, publisher, identifier, etc.
4. Deduplication module which filters out repeated references.
5. Mapping authors of extracted bibliographical references with members of the university staff. As we are interested only in scientific contribution by university researchers we decided to limit the scope of bibliographical entries to those belonging only to staff members.

We should point out that the first three modules address different types of classification problems and, henceforth, they will be called classification components.

### 4.1 General implementation of classification components

During the system development, we explored two approaches: rule-based and machine learning-based. Rule-based approaches usually give high precision but

rather low recall. In addition, their development is time-consuming as rules need to be manually produced, making them domain-dependent and difficult to adapt to other domains. In contrast, machine learning approaches usually ensure high recall and easy portability to other domains, with the downside that a large amount of annotated data is required for training in order to achieve sufficient accuracy. As we are mostly interested in high recall rather than in very precise results, we decided to adopt a machine learning approach and to increase accuracy by increment of training data and experimenting with different sets of features and machine learning techniques.

The development of classification components requires the following steps:

*1. Establishing a set of suitable machine-learning techniques.*

Machine-learning methods must be chosen according to the task they are solving. Selection of web pages containing publications is a regular classification task, while the extraction of bibliographical fields can be seen as a sequence labeling procedure. The latter should be treated differently applying methods, which take into account not only the features of the element but also its neighborhood and location in a sequence.

*2. Establishing a set of relevant features.*

Due to the same subject of the three tasks we use a similar set of features while constructing corresponding classifiers:

- **Named Entities**, such as PERSONS, LOCATIONS, ORGANIZATIONS and DATES that were tagged using Annie<sup>16</sup> parser, a part of GATE<sup>17</sup> package.

- **Staff list**. A list of all university members of research staff was collected and stored in the database. The list is regularly renewed in order to capture any alterations in staff complement.

- **Sherpa/Romeo publishers** - a list of publishers stored in Sherpa/Romeo database.

- **Sherpa/Romeo journals** - a list of journals stored in Sherpa/Romeo database.

- **Presence of year**. This feature indicates whether a corresponding element contains a year.

- **Publication triggers**. We built different lists of triggers: for example, header triggers (the most frequent words that occur in <h> tags of files containing publications), publication triggers (words appearing in a bibliographical entry itself), citation triggers (words appeared in citation of an entry).

- **Orthographic features**, which contain capitalization, digits, punctuation marks, etc. This feature was only used for implementation of the IE module.

- **Parts of speech (POS)**. Preliminary experiments revealed many cases when automatically annotated bibliographical fields ended with articles, conjunctions or prepositions. In order to correct this situation we incorporated these

---

<sup>16</sup> <http://gate.ac.uk/ie/annie.html>

<sup>17</sup> <http://gate.ac.uk/>

parts of speech into the IE module.

### *3. Experimenting with feature adjusting and classifier selection.*

It is a well-known fact that not all the features contribute positively into the final result of the classification and that it is a very important to select a set of features that gives the highest accuracy. Moreover, a great amount of noisy features increases the time costs without any improvement in the system performance. In Section 5 the experiments conducted with different numbers of features are described and discussed.

## **4.2 Page classifier**

Web pages with publications can have different formats: HTML, PDF or just plain text. It is obvious that not only the content but the structure of a document must be used in order to locate bibliographical entries. Concerning HTML format we distinguish 3 types of structured text that can be useful for revealing publications:

- Metadata - text appeared in meta tags like <keywords>, <title> and <description>. We also consider anchor text as it usually contains an explicit description of document content.

- Headers - text contained in header tags: <h1>, <h2>, etc. The text between these tags should be processed separately because if words like “publications” or “books” appear in headers the possibility that the file contain bibliographical entries is much higher than if those words occur in another part of a document.

- Content - the rest of a document.

As far as PDF and plain text formats are concerned, we only consider the content of documents due to absence of structural elements for these formats. .

The choice of machine learning methods was mostly defined by a set of methods contained in WEKA<sup>18</sup> - a well-known open source software implementing a number of machine learning algorithms for data mining. We employ 5 different methods: JRIP (rule learner algorithm described in [2]), Naive Bayes, Decision table (a simple decision table majority classifier [3]), J48(C4.5 decision tree classifier) and SMO (Support vector machine classifier as described in [4]).

## **4.3 Record classifier and IE module**

When classifying document records it is very important to incorporate information about a document structure together with features of separate records. One such structural element that can reveal enumerations or a list of equivalent records is an HTML-tag of a record. As an order of records can also help to correctly determine the start and end point of a publication list, we decided to apply a finite state transducer - Conditional Random Fields (CRF) [5] that are proved to be one of the most effective methods for sequence labelling tasks. In

---

<sup>18</sup> <http://www.cs.waikato.ac.nz/ml/weka/>

order to implement CRF-based IE module an open source Mallet package [6] was used.

The IE module needs to identify 5 different metadata types: author, title, citation, date and publisher (extraction of identifiers was not carried out due to the lack of training data). As bibliographical reference represents a logical consequence of metadata tags, the use of CRF is indisputable.

One of the important parameters of CRF classifier is a Markov order, characterising a number of previous elements which can influence the state of a pending one. In Section 5 we experiment with different values of Markov orders and show the effect of this on the evaluation results.

#### 4.4 Deduplication module

Once bibliographical entries are extracted and stored in the database, the deduplication analysis has to be carried out. References can be repeated in different web pages of the university, moreover, some spelling mistakes can occur, which makes the deduplication process more complicated. In our implementation the deduplication module is based only on the first author's name and a title, which means that if they are equal or similar the corresponding references are considered to be equal. For measuring similarity we use Levenshtein edition distance: it was applied at word level for titles and at letter level for author's names. Different thresholds for both titles and names were established in order to allow spelling mistakes to be introduced into the records.

The process of comparing database entries is a very time-consuming, moreover, in the case of similar references the problem of choosing one of them as the target reference arises. In order to make the process more efficient we employed Lucene<sup>19</sup> search engine that indexes first author's name and publication title for each database entry. Each new entry is compared with the stored ones and the number of times it appears is counted. At the end if there are several similar references we choose the one with the larger number of occurrences. In case of an equal number of occurrences the entry with the longer title is selected. This process ensures that we always obtain more complete titles without abbreviations and more complete names instead of only initials.

#### 4.5 Mapping authors with members of the university staff

Once all repeated references are removed researchers belonging to the university staff should be associated with their publications. Obviously one publication can be connected with different members of staff if it has multiple authors. Our method of mapping authors with members of staff consists in comparing author's last and first names and if only initials are available the first names are shortened to compare only first letters. One of the problems that can occur during this procedure is an ambiguity problem caused when two or more members of staff have the same name. In this case we associate a bibliographical entry with all of them, giving the authors responsibility to clarify the ambiguity.

<sup>19</sup> <http://lucene.apache.org>

## 5 Evaluation of the AIR core module

Evaluation of the AIR core module was carried out separately for each of its classification components. In all experiments we used cross validation with number of folds equal to 5 in order to obtain a reliable estimation of the system performance.

### 5.1 Evaluation of the page classifier

The data consists of about 1850 manually annotated web pages, 150 of them containing publications. Table 1 gives the results obtained by all employed methods. The best performance was achieved by the JRIP method which outperformed other methods both in recall and F-score reaching 90% of recall. Although 90% is quite a good result it is not sufficient for our purposes because we lose 10% of relevant web pages, which means that 10% of authors are missed by the system. In order to avoid missing relevant data we decided to skip this stage in the final implementation of the AIR system, assuming that all web pages contain publications and passing them directly to the record classifier. Due to rather small size of the university web site (approximately 80 000 web pages) the classification of records of all university web pages is not a time- or resource-intensive process.

**Table 1.** Performance of the page classifier

N	Method	Precision	Recall	F-score
1	JRIP	0.864	<b>0.901</b>	<b>0.882</b>
2	Decision table	0.863	0.801	0.831
3	Naive Bayes	0.43	0.894	0.583
4	SMO	0.853	0.823	0.838
5	J48	<b>0.870</b>	0.809	0.838

### 5.2 Evaluation of the record classifier

The data consists of 55 web pages, total number of records is equal to 4200, 520 of which are bibliographical entries. We conducted our experiments for different Markov orders: 1, 2, 3. The results of the experiments are shown in Table 2.

**Table 2.** Performance of the record classifier

Markov order	Precision	Recall	F-score
1	0.929	<b>0.914</b>	<b>0.919</b>
2	0.929	0.898	0.910
3	0.929	0.859	0.888

The model based on bigrams obtained the best performance which means that knowledge about more distant neighbours of a bibliographical entry is not necessary and only brings noise into input data.

The overall performance of this component can be estimated as good although we have the same problem as for the previous module: in spite of quite high recall (91.4%) the risk of skipping bibliographical references still exists. In any case, as the classification is conducted at record level, we can miss only separate entries, not the whole pages.

### 5.3 Evaluation of IE module

The data represents 520 publication references manually annotated with bibliographical metadata. In Tables 3-4 results of evaluation for Markov orders equal to 1 and 2 respectively are presented. Our baseline corresponds to the model where a set of standard features used for previous procedures was used: words, staff list, citation triggers, presence of year, etc. We compare our baseline with richer models adding also POS and orthographic features. In order to estimate statistical significance of differences in performance shown by different models the results are given with their confidence intervals calculated for  $\alpha = 0.05$  (confidence level equal to 0.95). Comparison of the results for different Markov

**Table 3.** Performance (F-score) of IE module for Markov order 1

Model	Author	Date	Title	Citation	Publisher
baseline	$0.968 \pm 0.007$	$0.872 \pm 0.033$	$0.929 \pm 0.008$	$0.878 \pm 0.009$	$0.438 \pm 0.047$
bas+POS	$0.968 \pm 0.007$	$0.859 \pm 0.034$	$0.918 \pm 0.009$	$0.860 \pm 0.010$	$0.379 \pm 0.046$
bas+ortho	$0.984 \pm 0.005$	$0.865 \pm 0.034$	$0.940 \pm 0.008$	$0.895 \pm 0.009$	$0.538 \pm 0.047$
bas+POS+ortho	$0.984 \pm 0.005$	$0.870 \pm 0.033$	$0.942 \pm 0.008$	$0.899 \pm 0.009$	$0.556 \pm 0.047$

**Table 4.** Performance (F-score) of IE module for Markov order 2

Model	Author	Date	Title	Citation	Publisher
baseline	$0.974 \pm 0.006$	$0.860 \pm 0.034$	$0.928 \pm 0.009$	$0.877 \pm 0.009$	$0.379 \pm 0.046$
bas+POS	$0.971 \pm 0.007$	$0.857 \pm 0.035$	$0.922 \pm 0.009$	$0.870 \pm 0.010$	$0.352 \pm 0.046$
bas+ortho	$0.975 \pm 0.006$	$0.869 \pm 0.033$	$0.944 \pm 0.008$	$0.899 \pm 0.009$	$0.562 \pm 0.047$
bas+POS+ortho	$0.982 \pm 0.005$	$0.878 \pm 0.033$	$0.956 \pm 0.007$	$0.918 \pm 0.008$	$0.616 \pm 0.046$

orders shows a slightly better performance of the more complex model, which is statistically significant for the fields “title”, “citation” and “publisher”.

As far as feature contribution is concerned it is obvious that orthographic features improve the model performance especially for the fields “author”, “citation” and “publisher” as these are the fields with the greatest number of

punctuation marks. The positive contribution of POS tags is not that evident, they even worsen the results for the majority of fields if they are taken without orthographic features. The best result obtained for Markov order equal to 2 and the richest set of features demonstrates slight contribution of POS tags, mostly for extraction of citations and publishers.

In general, the evaluation of the IE module shows its good performance with very high F-score obtained for extraction of authors and titles. Low F-score for the field "publisher" can be explained by the lack of training data as many authors do not introduce this information while posting their publications. Another difficulty of publisher's extraction is caused by the presence of geographical entities (or, more precisely, cities) in their names. Therefore, they can be easily confused with citations.

## 6 Conclusions and future work

The presented work introduced a semi-automatic system for archiving institutional repositories that was developed at the university of Wolverhampton to foster the dissemination of research output produced by the university staff. Manual population of institutional repositories is an extremely time- and resource-intensive process and, therefore, their growth is restricted by limits dictated by human resources. The AIR system aims to automate the data archiving process, so that librarians will be involved only at the final stage of the system for data verification and correction.

The main challenge the system copes with is caused by the need to deal with unstructured Web data that could have rather diverse formats. The evaluation results demonstrate high system performance and encourages us to improve the system further in order to make it more convenient for users.

As future work, we plan to extend the volume of the training data set reusing already verified references obtained at the end of the system cycle. We expect to improve the system performance not only because of the growth of training data but also due to the fact that the accumulation of references' formats used at the university adjusts the system to the university standards and requirements. Another direction of our future research will be related to conducting more experiments with other Machine Learning algorithms and other feature sets in order to raise the system accuracy.

Better access to the bibliographical metadata is currently being explored using the QALL-ME framework<sup>20</sup>, a domain independent architecture that can be used to easily implement question answering systems. By using the framework we envisage to be able to give users access to the contents of the database in a natural way, by answering questions about publications such as "What authors published papers in the Journal of Natural Language Engineering in 2002?".

---

<sup>20</sup> <http://qallme.sourceforge.net/>

## Acknowledgements

This work has been partially supported by the Joint Information Systems Committee (JISC)<sup>21</sup> through the AIR project. Some system modules have been developed in the framework of the project QALL-ME<sup>22</sup>, which is a 6th Framework Research Programme of the European Union (EU), contract number: FP6-IST-033860 and the TEXT-MESS project. The authors also want to acknowledge Javier Espinosa de los Monteros for his large contribution to the development of the Web interfaces.

## References

1. Haase, P., Broekstra, J., Ehrig, M., Menken, M., Plechawski, M., Pyszlak, P., Schnizler, B., Siebes, R., Staab, S., Tempich, C.: Bibster - a semantics-based bibliographic peer-to-peer system. In: Proceedings of the Third International Semantic Web Conference. (2004) 122-136
2. Cohen, W.W.: Fast effective rule induction. In: Proceedings of the Twelfth International Conference on Machine Learning, Morgan Kaufmann (1995) 115-123
3. Kohavi, R.: The power of decision tables. In: Proceedings of the European Conference on Machine Learning, Springer Verlag (1995) 174-189
4. Platt, J.C.: Fast training of support vector machines using sequential minimal optimization. In: Advances in kernel methods: support vector learning, MIT Press (1999) 185-208
5. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc. (2001) 282-289
6. McCallum, A.K.: Mallet: A machine learning for language toolkit. (2002) <http://mallet.cs.umass.edu>.

---

<sup>21</sup> <http://www.jisc.ac.uk/>

<sup>22</sup> <http://qallme.itc.it/>