RIDIRE-CPI: an Open Source Crawling and Processing Infrastructure for Web Corpora Building

Alessandro Panunzi, Marco Fabbri, Massimo Moneglia, Lorenzo Gregori, Samuele Paladini

University of Florence

alessandro.panunzi@unifi.it, marco.fabbri@drwolf.it, moneglia@unifi.it, loregreg@gmail.com, samuelepaladini@drwolf.it

Abstract

This paper introduces the RIDIRE-CPI, an open source tool for the building of web corpora with a specific design through a targeted crawling strategy. The tool has been developed within the RIDIRE Project, which aims at creating a 2 billion word balanced web corpus for Italian. RIDIRE-CPI architecture integrates existing open source tools as well as modules developed specifically within the RIDIRE project. It consists of various components: a robust crawler (Heritrix), a user friendly web interface, several conversion and cleaning tools, an anti-duplicate filter, a language guesser, and a PoS tagger.
The RIDIRE-CPI user-friendly interface is specifically intended for allowing collaborative work performance by users with low skills in web technology and text processing. Moreover, RIDIRE-CPI integrates a validation interface dedicated to the evaluation of the targeted crawling. Through the content selection, metadata assignment, and validation procedures, the RIDIRE-CPI allows the gathering of linguistic data with a supervised strategy that leads to a higher level of control of the corpus contents. The modular architecture of the infrastructure and its open-source distribution will assure the reusability of the tool for other corpus building initiatives.

Keywords: web corpora, crawling, LRs construction.

1. Introduction

1.1 Web corpora: a brief state of the art

In recent years, different practices about using the web as linguistic data source broke into the scientific community. The idea to use the web itself as a corpus "surrogate" gave rise to an interesting discussion between its supporters and detractors. The building of "reference" corpora starting from web materials might be considered inadequate due to the lack of representation of the "language of the web" and the low level of control on data (Ide et al., 2002). On the other hand, the web can be considered the largest linguistic corpus in the world (Kilgariff & Grefenstette, 2003).
Due to the enormous growth of the Internet, the language data collected in the web is not exclusively "the language of the web". As a matter of fact, texts from many different sources can be found therein: literature, newspapers, academic and bureaucratic prose. Therefore, a collection of its linguistic data can exploit the huge richness and variety of the sources. Its balancing is not a problem related to the data sources, but, rather, depends on the methods of collection.
Different kinds of projects have been carried out in order to exploit the language data that populates the web. Some of them focused on the direct exploitation of the Internet through search engine techniques (e.g. WebCorp, Renouf et al., 2007). Others were interested in massive language collections (with an almost absence of further control and processing of data) for strict computational purposes (e.g. Clarke et al., 2002, 53 billion words).
Still others tried to extract more controlled corpora from the web, using methods that involve various steps of cleaning and tagging (e.g. CUCWeb, Boleda et al., 2006; Sharoff, 2006). Among these latter ones, the WaCky initiative (Baroni et al., 2009) represents the main effort by a group of linguists to produce mega corpora suitable for linguistic research. Four very large corpora have been collected for English, French, German and Italian, each one between 1.6 and 2 billion tokens. Within this initiative, an interesting and fully automated method has been designed in order to assure a certain degree of balancing and representativeness, exploiting corpus-based frequency lists and basic dictionaries to perform the web queries from which the data collection derives.

1.2 RIDIRE: project and tools

An even more controlled method has been developed in the RIDIRE project (Moneglia & Paladini, 2010), which aims to build a web-derived repository for the Italian Language (2 billion words), with a specific corpus design. The whole resource is composed of a set of sub-corpora, each one collecting texts of a specific functional or semantic domain (news, law/administration, economy, literature, fashion, architecture/design, food, sport, religion, fine arts, cinema, music).
The gathering of linguistic data for each sub-corpus requires a targeted crawling strategy. The content restrictions are directly undertaken by the corpus collector who selects, from an "expert" point of view, a starting webography that is supposed to be representative of a given domain.

This paper describes the tool that has been developed within the RIDIRE project for the crawling and the processing of the web resources: the RIDIRE Crawling and Processing Infrastructure (henceforth, RIDIRE-CPI). This tool will be delivered as an open source package and will be available for the development of other corpora.
The infrastructure can be used for the building of domain-specific corpora for whatever computational purpose. Moreover, its user-friendly interface is specifically intended for allowing collaborative work performed by users with low skills in web technology and text processing.
The infrastructure has a modular architecture (see figure 1), that consists of the following main components:

1) a web crawler;
2) a web interface for crawling management and validation;
3) conversion tools;
4) HTML cleaner tools;
5) anti-duplicate filters;
6) a language guesser;
7) a PoS tagger.

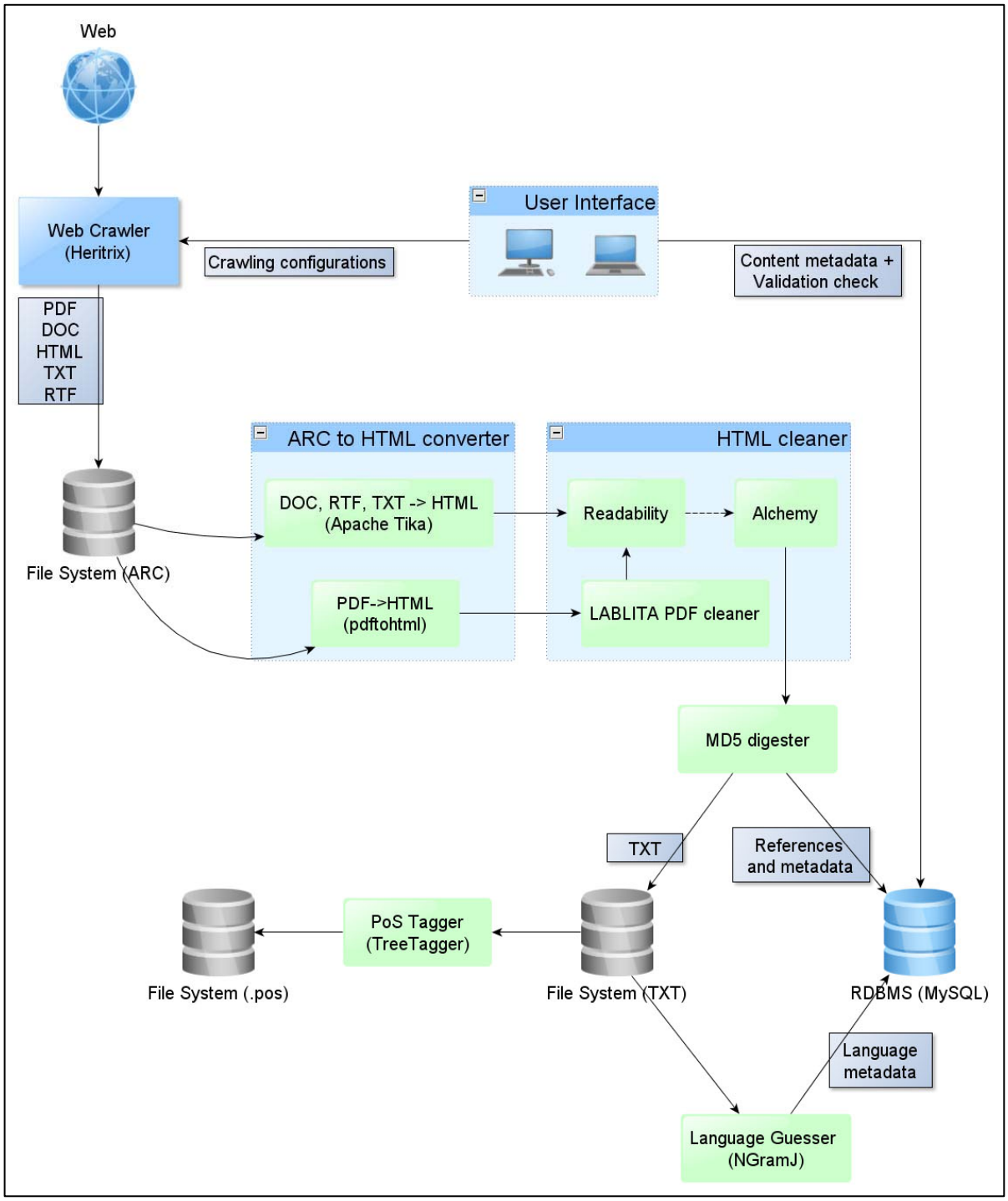

Figure 1: RIDIRE-CPI architecture

1.3 Infrastructure general design

RIDIRE-CPI is based on a JBoss/Seam application designed and developed expressly for this project. The application is written in Java and follows the JavaEE specifications.
The infrastructure provided by the JBoss/Seam framework allows an easy integration of repositories stored on RDBMSs, Business Logic and web interfaces. At this moment the application exploits JBoss 5.1, Seam 2.2.0 and Oracle Java 1.6.20.
The Business Logic inside the application coordinates and controls several tasks, namely:

- the crawling process (see paragraph 2);
- the mapping process (see paragraph 3);
- the user interaction via web interface.

Apart from the web interface, that is built "inside" the application (actually, following JavaEE specifications, Java code and web resources are clearly separated in different packages), the other tasks are performed by external "actors" (tools, applications, programs, web services) that run independently from the main application. This loose design gives great flexibility and allows the substitution of one or more of the external tools with little effort. Besides this, an error, a malfunction or simply an upgrading of one of the actors doesn't stop the whole system. For example, if some jobs are in crawling state (i.e. web resources are being collected) and an upgrade to the main application is needed, the administrator of the server can simply shutdown the JBoss application and perform the necessary upgrading task, which could take some time, while running jobs continue working; then, after restarting, the application will fetch updated data from the crawler. This loose binding operates in both ways, as, for example, the main application will work without complaining if the crawler is not running: the operations, activities and data that the crawler provides simply won't be available to users.

2. Crawling process

2.1 Heritrix

The crawling activity in RIDIRE-CPI is based on Heritrix (version 3.1.1), an open-source (Apache license ver. 2.0) and reliable crawler that is used extensively worldwide. Heritrix is a Java application based on the Spring framework, and it was chosen among other open-source crawlers (e.g. Nutch) for its great degree of customization and because of the number of applications using it at the time of design.
Heritrix copes with the huge amount of technical problems that the web crawling process has, such as:

- politeness; the crawler should not generate too much traffic on a crawled host;
- robots.txt; the crawler should respect specifications about resources' crawling permissions;
- missing resources; web pages may have broken links;
- loops; some web sites can be designed with link loops;
- multiple URLs for same resource; the crawler should not download the same resource twice;
- load balancing; if multiple jobs have to be run, the crawler takes care of distributing CPU resources and bandwidth so each one of them can work.

2.2 Web interface for crawling

RIDIRE-CPI configures Heritrix *via* its REST API, by means of the web interface. The crawling activity is structured in "jobs", i.e. fully configured crawling sessions. To configure a crawling session, the user has to specify three sets of parameters in the interface (figure 2). First, the user selects the *seeds* i.e. the set of URLs from which the activity starts. Heritrix saves the seeds in the "URL queue". Given this, the crawling activity proceeds along the following recursive steps:

1) the crawler accesses the web page relative to the first URL in the queue;
2) it extracts all the links and saves them in the "URL queue";
3) it downloads the web page content and saves it into the file system;
4) it goes back to the first step.

The second set of parameters given to the crawler is the formats (MIME types) of the resources that the user wants to download (Heritrix can discriminate among the MIME types of processed resources, and store only the ones chosen in the configuration). In addition to HTML, RIDIRE-CPI is able to process documents in TXT, RTF, DOC, and PDF. This feature is crucial, since many linguistically relevant resources in the web are not contained in web pages, but in documents with various formats.
The third set of parameters provides an additional strategy for the content selection of a web site. In this step, the user selects and/or discards the "resources" (i.e. web pages and documents in various formats) exploiting the regularities of the URLs which refer to them. The given parameters consist of two sets of strings that can be expressed either as enumerated lists or as regular expressions. The user gives this information to the crawler after the analysis of the web site URL structure. The two sets of strings specify:

- which URLs the crawler has to add to the queue ("URL to be navigated");
- which resources the crawler has to download to the file system ("URL to be saved").

3. Mapping process

3.1 Conversion to HTML and cleaning

It is well known that, for the aim of building a corpus that is adequate for linguistic research, the crawled data has to be processed by a complex procedure that includes, for instance, text cleaning, duplicate removal, and POS-tagging (Baroni et al., 2009).

To this aim, RIDIRE-CPI implements the "mapping" process, which corresponds to an automatic processing pipeline (i.e. a series of cascading procedures) on the downloaded resources in order to extract the running text that will constitute the corpus itself.

For each terminated job, the RIDIRE-CPI stores an ARC archive in the file system that contains the downloaded resources in different formats. To proceed to the subsequent steps, all the resources are first converted into HTML. For this task, several tools are used depending on the input format:

1) "Apache TIKA" for DOC, RTF and TXT conversions;
2) "pdftohtml" plus an ad hoc cleaner (LABLITA PdfCleaner) for the PDF conversion.

The conversion is embedded in a common Java application (that can be executed on the command line, passing some parameters) that runs outside the main JBoss application. Again the loose coupling is made so that if the conversion ends in error or, worse, crashes, the main application is not affected.

Apache TIKA is used as a programming library by the conversion application. On the contrary, pdftohtml and PdfCleaner are, in turn, used as external programs: the first tool is used for the main conversion, while the second has been specifically developed for discarding tables, indexes, page numbers and notes from the converted HTML file.

After the conversion, the text cleaning is performed. Web pages, as is well known, contain text that is not relevant for the constitution of a corpus i.e. advertising, navigation menus, disclaimers, credits etc. (the so called "boilerplate"). This kind of text is removed using two external tools:

1) Readability;
2) Alchemy API.

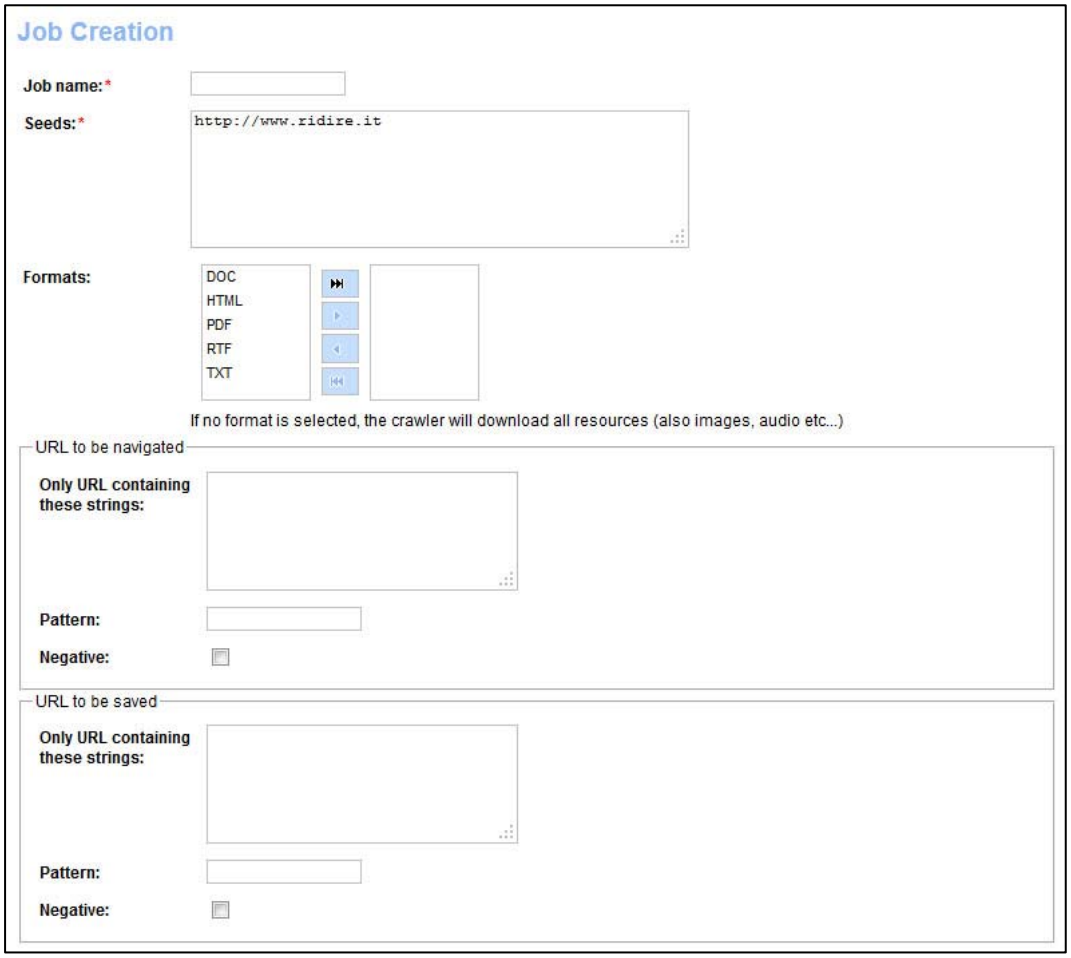

Figure 2: Job configuration interface

Both are embedded in a Java cleaning application controlled by the JBoss main application.
Readability is normally used as a browser plugin to obtain more readable texts from web pages filled with uninteresting banners and links around the main text. Readability is actually a Javascript program that is launched when the user presses the plugin icon. In RIDIRE this behavior is simulated using HTMLUnit, a Java programming library commonly used to test web pages. As HTMLUnit contains a Javascript engine (Mozilla Rhino), in RIDIRE it is used to execute the Readability script against the HTML resource as in a common browser.
Even if Readability has been evaluated as the best HTML cleaner among the ones taken into account, it is possible that it won't yield results or will output an error. The Alchemy API has been chosen as a second chance cleaner, and is a web service that exposes a REST API and provides a Java (and other programming languages) Software Development Kit to embed Alchemy calls in Java code. The SDK is used by the cleaning application.

3.2 Mapping the resources onto the RDBMS

The plain text documents that output from the cleaning stage are then processed by a simple MD5 digester to get their signature. This signature is used to name the resource and acts as the first anti-duplication system. Before storing the resource, the application checks the DB to see if another resource with the same signature exists; if so, the new resource is discarded. If the resource is actually new, the resulting file is written to the file system. The mapping process will also integrate a near duplicate remover (to be implemented) based on a keyword extractor (Panunzi et al., 2006).
After this, and before a crawled resource can be created in the database (RDBMS), the language of the resource must be retrieved. It's not possible to assign this to a job during the definition, or to trust language information provided by web servers or HTML pages, since they are highly unreliable. Thus, a statistical language guesser is used: NGramJ. NGramJ is an open-source (LGPL) language guessing Java library. It's very fast, complete, and, because of its reliability, is directly embedded in the main application.
With this new information the JBoss application can create a so-called CrawledResource, which is simply a record in the RDBMS that stores metadata information regarding the resource. For metadata storage RIDIRE uses MySQL (version 5.1).
The last phase of the mapping procedure is the part of speech tagging of the plain text resource. PoS tagging is performed by TreeTagger, which is run by the main application as an external executable. The PoS-tagged file is directly created in the correct location by TreeTagger.

4. Supervision and validation

At the end of the mapping process, each resource has its own record in the RDBMS. Such a record contains:

- the original URL of the resource;
- the MIME type of the resource;
- the reference to the extracted text in the file system;
- the reference to the PoS-tagged text in the file system;
- the language of the text;
- the number of words (excluding non-word tokens) of the PoS-tagged text;

The RIDIRE-CPI web interface shows all of this information to the user, who can use it to verify the results of the crawling job.
Through the interface, the user is also provided with control functions on the downloaded resources. Exploiting the metadata on word numbers, he can choose to delete from his job all of the texts that contain less than 100 words, with the aim of avoiding an over-fragmentation of the resulting corpus. He can also assign content metadata to the resources, which can be used for creating and balancing sub-corpora within a larger collection.
Moreover, RIDIRE-CPI integrates a validation interface dedicated to the evaluation of the targeted crawling. The validation procedure creates a random sample of the resources contained in a job. The user can check whether the job resources are adequate with respect to the corpus design or content restrictions. A job can be considered “valid” if it contains non adequate resources under a certain percentage.
Through the content selection, metadata assignment, and validation procedures, the RIDIRE-CPI allows the gathering of linguistic data with a supervised strategy that leads to a higher level of control. The modular architecture of the infrastructure and its open-source distribution will assure the reusability of the tool for other corpus building initiatives.

5. References

Baroni, M., Bernardini, S., Ferraresi, A. and Zanchetta, E. (2009). The WaCky Wide Web: A Collection of Very Large Linguistically Processed Web-Crawled Corpora. *Language Resources and Evaluation* 43(3), pp. 209--226.

Boleda, G., Bott, S., Meza, R., Castillo, C., Badia, T. and Lopez, V. (2006). CUCWeb: A Catalan corpus built from the web. In A. Kilgarriff & M. Baroni (Eds.), *Proceedings of the 2nd International Workshop on the Web as Corpus*. East Stroudsburg (PA): ACL, pp. 19--26.

Clarke, C., Cormack, G., Laszlo, M., Lynam, T. and Terra, E. (2002). The impact of corpus size on question answering performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York (NY): ACM, pp. 369--370.

Ide, N., Reppen, R. and Suderman, K. (2002). The American National Corpus: more than the Web can

provide. In *Proceedings of the 3rd Language Resources and Evaluation Conference, LREC 2002*. Paris: ELRA, pp. 839--844.

Kilgarriff, A. and Greffenstette, G. (2003). Introduction to the Special Issue on Web as Corpus. *Computational Linguistics* 29(3), pp. 1--15.

Moneglia, M. and Paladini, S. (2010). Le risorse di rete dell'italiano. Presentazione del progetto "RIDIRE.it". In E. Cresti & I. Korzen (Eds.), *Language, Cognition and Identity*. Firenze: Firenze University Press, pp. 111--128.

Panunzi, A., Fabbri, M. and Moneglia, M. (2006). Integrating Methods and LRs for Automatic Keyword Extraction from Open-Domain Texts", in *Proceedings of the 5th Language Resources and Evaluation Conference, LREC 2006*. Paris, France: ELRA, pp. 2205--2208.

Renouf, A., Kehoe, A. and Banerjee, J. (2007). WebCorp: an integrated system for web text search. In C. Nesselhauf, M. Hundt & C. Biewer (Eds.), *Corpus Linguistics and the Web*. Amsterdam: Rodopi, pp. 47--67.

Sharoff, S. (2006). Creating general-purpose corpora using automated search engine queries. In M. Baroni & S. Bernardini (Eds.), *Wacky! Working papers on the Web as Corpus*. Bologna: Gedit, pp. 63--98.

Project sites and Tools

Alchemy API. http://www.alchemyapi.com/
Apache Tika. http://tika.apache.org/
CucWeb. http://ramsesii.upf.es/cucweb/about.en_US.htm
Heritrix. http://crawler.archive.org/
HTMLUnit. http://htmlunit.sourceforge.net/
Leeds collection of Internet corpora. http://corpus.leeds.ac.uk/internet.htm
NGramJ. http://ngramj.sourceforge.net/
pdftohtml. http://pdftohtml.sourceforge.net/
Readability. http://www.readability.com/
TreeTagger. http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/
WaCky. http://wacky.sslmit.unibo.it/doku.php
WebCorp. http://www.webcorp.org.uk/

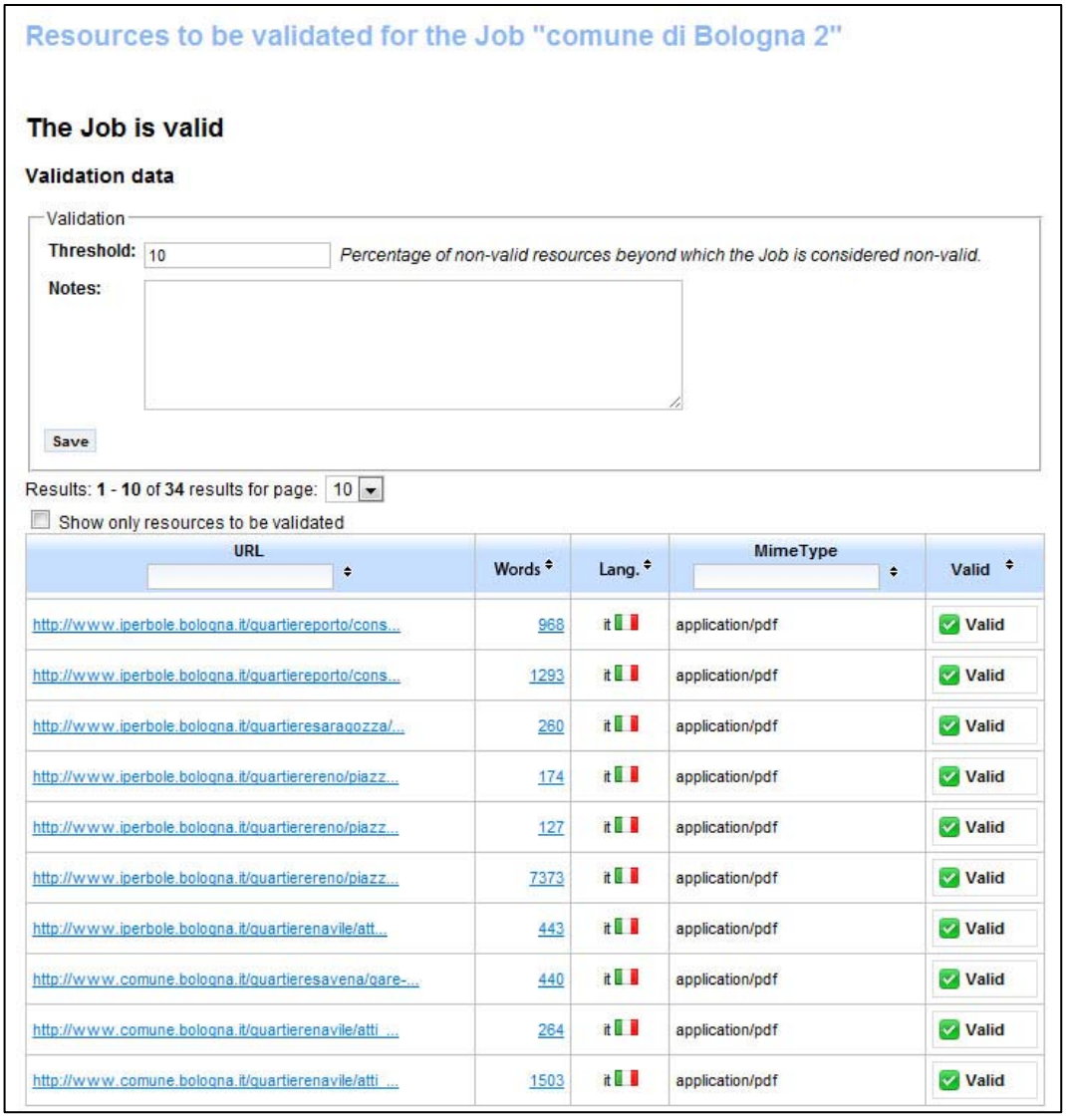

Figure 3: Validation interface