ASSIST project - internal report ASSIST-D3, Davy Weissenbacher, Brian Rea, Sophia Ananiadou, November 2008

ASSIST, a Specialized Search Engine for SOCIAL SCIENCES Documents

Tutorial

1 How to use the Beta-Version of ASSIST search engine

The Beta-Version of the ASSIST[1] search engine has been conceived as a web-based system. To use it, the user needs to have an active internet connection to connect to our server and to have installed a browser such as Internet Explorer or Mozilla-Firefox as well as a recent java virtual machine.

The ASSIST search engine[2] has been duplicated and deployed twice to work on two different corpora. The first search engine, which we call the NCESS search engine, is designed to help a sociology researcher in frame analysis work. The NCESS search engine allows sociologists to retrieve relevant documents in a corpus composed of 4000 newspapers extracted from the LexisNexis database. The second search engine, the EPPI search engine, is designed as a specialized search engine for the education domain. The EPPI search engine allows a user interested in education to browse 1300 documents from different education web sites.

To connect to the beta version of both search engines, the user has to type the following URLs in his/her browser:

- NCESS search engine: http://nactem3.mc.man.ac.uk:8080/ASSIST/ (access can be provided on request). We detail the process in the section 2.1 Identification interface.
- EPPI search engine: http://nactem3.mc.man.ac.uk:8080/ASSIST-EPPI/ and connect with the user name: *EPPI* and the password: *access*.

The current version of both search engines has been tested on Mozilla-Firefox, Internet Explorer, and Epiphany. As other browsers have not been tested, we cannot guarantee complete compatibility with the Beta-version. The search engine uses an applet to visualize the results of the search as a cluster of documents (see section 2.4 Cluster visualization interface). To run correctly, a recent java virtual machine (version 1.6.0_06 and above) has to be installed on the client's system. Older versions are not supported, and the applet is not displayed.

2 Overview of the Beta-Version of the ASSIST search engine

ASSERT[3] is a system to produce Automatic Summarisation for Systematic Reviews using Text Mining. We have designed the ASSERT system for mass-media and educational documents, respectively the NCeSS search engine and the EPPI search engine. We present in this section the Beta-version of the NCeSS search engine. The EPPI search engine is used in an identical way.

The NCeSS search engine is composed of 5 different interfaces offering various ways to browse available documents. We describe in detail each component of the search engine.

1 Information about this project can be found at http://www.nactem.ac.uk/assist/
2 The core of this search engine used Lucene, details are provided in the website: http://lucene.apache.org/java/docs/
3 Information about this project can be found at http://www.nactem.ac.uk/assert/

2.1 Identification interface

The first interface is an identification interface. It invites the user to enter his/her user name and password in the appropriate fields. This identification takes place in the beta-version for testing purposes. In further versions this protection will be removed in the EPPI search engine to make it available to the public. The NCESS search engine will be still protected and used internally due to copyright issues with the LexisNexis corpus.

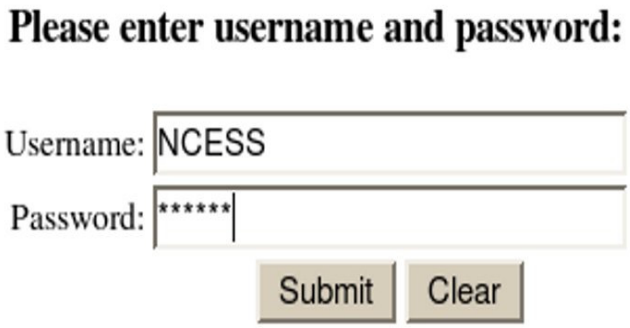

Illustration 1: Identification interface

Once the U*sername* and *Password* fields have been filled in, the user has to click on the Submit button to send the information. The server checks the validity of the information. If they are invalid or missing, the screen is displayed again, explaining why the information has been rejected, with empty fields awaiting new information. If the information is correct, the user is allowed to use the search engine through the query interface.

2.2 Query interface

This interface allows the user to write a query and send it to the search engine.

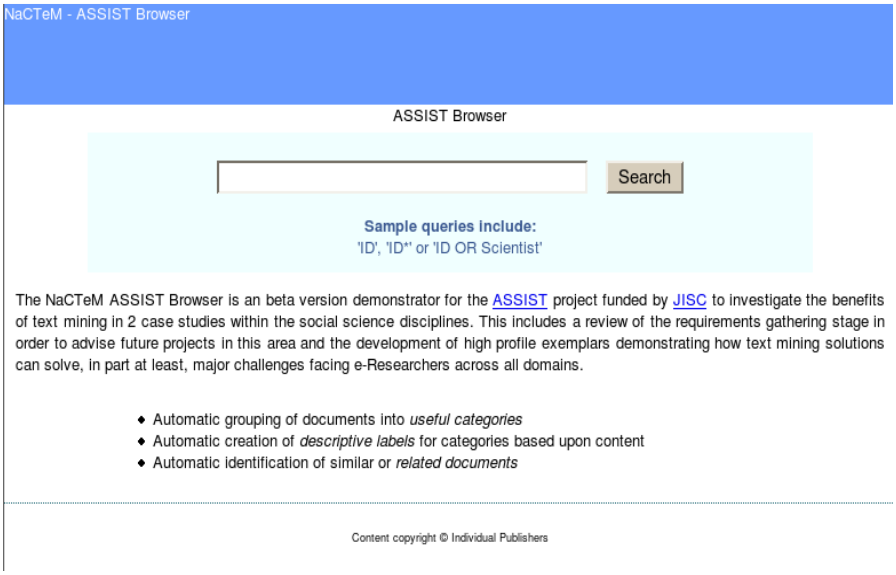

Illustration 2: Query interface

This interface provides a text field to write a query. By clicking on the Search button, the query is sent to the search engine to be processed, and the results are displayed in the next screen, the search result interface. Traditional and semantic operators can be inserted in the query to find the most relevant set of documents. We describe in the next table the operators currently available and the future operators implemented.

Query Type	Example	Comment
Word query		
One word plain text query	*passport*	The query will return all documents containing at least one occurrence of the word 'passport' in their contents.
multi-words plain text query	*ID card*	The query will return all documents containing at least one occurrence of the sequence formed by the multi-words. This query type will be available in the next version.
Wild-card character query		
?	*l?st*	Single character match; matches *lust, lest, lost last, etc.* Due to a search engine restriction this wild-card character cannot be in the first position of a word.
*	b*s	Multiple character match: matches bus, beens, business, *etc.*
Boolean queries		
AND	*word-A AND word-B*	Returns all the documents where *wordA* appears in the same document as *wordB*.
OR	*word-A OR wor-dB*	Returns documents containing word-A, *word-B* or both.
NOT	*word-A NOT word-B*	Returns all the documents where *wordA* appears and not the *wordB*
Semantic query		
source;	*source: Sun*	Using the predefined operator is possible to return the documents according to their *sources, authors, dates, etc.* For example the query *Authors: Wintour* returns all documents where Sir Wintour appears as author or co-author. The operators *subject* and *keywords* are not available for the NCeSS search engine. The operator *date* and *source* are not available for the EPPI search engine. In this version the operators not allow
authors:	*authors:Wintour*	
date:	*date: 2006*	
title:	*title: privacy*	

subject:	*subject: research*	any variation. Note that for some documents although expected pieces of information can be missing, e.g. authors field not fulfilled in the document.
keyword:	*keywords: education*	
term:	*term:id**	Using these operators is possible to specify words with specific properties. The operator *term* allows a research on multi-words wich are identified a concept of the domain, wich are called terms, and with NECanonical the words denoting the discourse objects refered as Named Entities. Currently the query function for multi-words is not implemented, consequently the multi-words terms can only be queried through the wild-card '*' as in the example. The next version will correct this limitation. One more operator will be added in next version, the *NEType* to precise the type of a discourse object *e.g. a city, a human, etc.*
NECanonical:	*NECanonical: London*	
Composed Query		
	*Date: 2006 AND title:ID**	It is possible to combine different operators in a query to express a precise idea like in the example, the query will return all documents written in 2006 containing *ID card, ID scheme, etc.* in their titles.

2.3 Search result interface

Once a well-formed query has been sent to the server, the search engine returns the most relevant documents for this query. The result is displayed in the search result interface.

This interface is composed of three parts. The first part is the first line of the interface (number 1 in Illustration 3). This part informs the user of the query submitted, in Illustration 3 'passport', and a hyperlink *'Visualize'* allowing the user to see the results through the cluster visualization interface described in the next subsection.

The second part is located on the right side of the screen (number 3 in Illustration 3). This presents a list of all pertinent documents for the query. The title of each document is a hyperlink that the user can click to access the full document as described in Section 2.5 Document visualization interface. The authors and the date of the document's publication are displayed under the title. These fields are currently semi-structured, *i.e.* the description of the full information is incomplete; the string 'September 3, 2005' is known to be a date, but its components, the month, the day, and the year, are not identified as such. This prevents precise requests with new operators (*e.g. 'month:september AND year:2005*). When these fields are not available, the value *'unknown'* is displayed. Under them a list of a maximum of 5 terms is given as an overview of the document content. The terms are

calculated using a Text Mining tool, called Termine [Frantzi *et al.* 00]. Termine associates a score according to the importance of the term in the document. The list is presented in a decreasing order of the termine score. A threshold on this score determined by observation on our corpus filters out unimportant terms from the list.

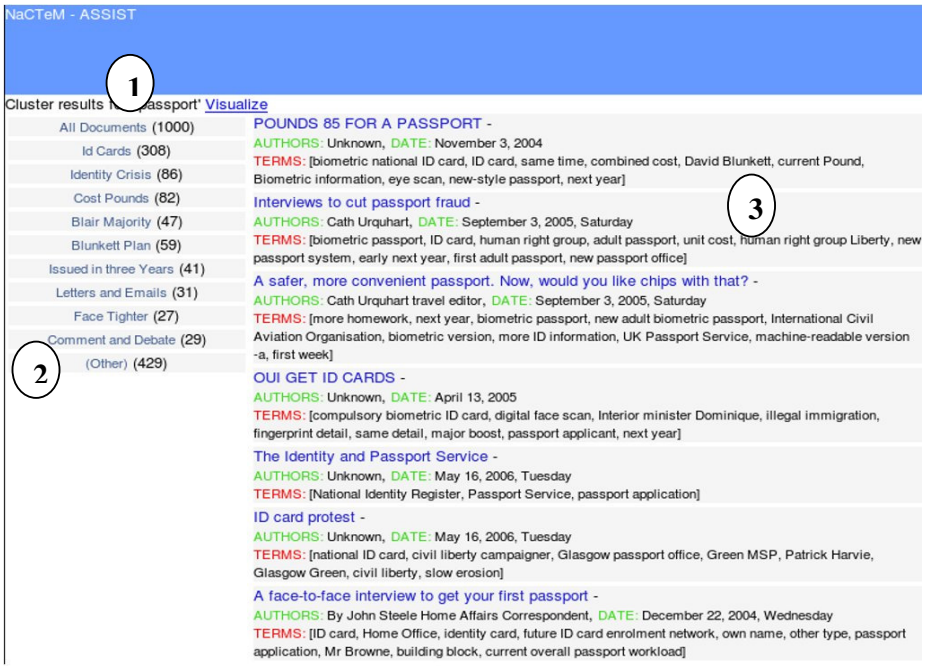

Illustration 3: Search result for the query 'passport'

The third part, located in the left part of the screen (number 2 in Illustration 3), divides into clusters the set of documents returned for the query. The documents are clustered according to the topic they share. A cluster receives a title computed automatically by the search engine[4]. This title is the best representative term (*i.e.* roughly speaking the more frequent term) for the topic of the cluster. The first title *'All Documents'* sums up the number of documents found for this query. Currently we have limited the size of the search result to the first 1000 documents to reduce the calculation time and facilitate the implementation, but this limitation may be removed in further versions. The last title *'Other'* is the cluster of documents which couldn't be inserted in an existing cluster. The titles are hyperlinks. Clicking on them changes the selection of documents in the second part of the screen: only documents belonging to the selected cluster are displayed.

2.4 Cluster visualization interface

To access the cluster visualization interface, click on visualize in the Search Result Interface. This interface is a graphical representation of the clusters[5].

The most important part of this interface is located in the right-hand part of the screen (number 3 in Illustration 4). This part displays a graphical representation of a set of clusters. Each cluster gets a unique colour. Each document contained in the cluster is represented by a yellow bubble. When the mouse points to a bubble, the title of the document and the URL of the document are displayed in the field under the cluster (in our illustration the URL of the document is *LNDocument3863.xml* and its title '*Peers block identity cards over cost*'). This view is a representation of the intersections

[4] The calculation is realized using the Lingo algorithm [Osinski, 03].

[5] The software used for this graphical representation is Aduna wich can be downloaded from the website: http://www.aduna-software.com/technologies/clustermap/overview.view

between the sets of clusters. When a document belongs to two or more clusters, the coloured areas representing the cluster are joined by a new link, and the document is put in their intersection (in the illustration only one document is a member of all three clusters).

The part numbered 1 in Illustration 4, as in the Search result interface, informs the user of the query submitted. The second line is a list box to set the scale of the graphical representation of the cluster. Choosing a percentage in the list box, the user changes the size of the clusters displayed in the part 3.

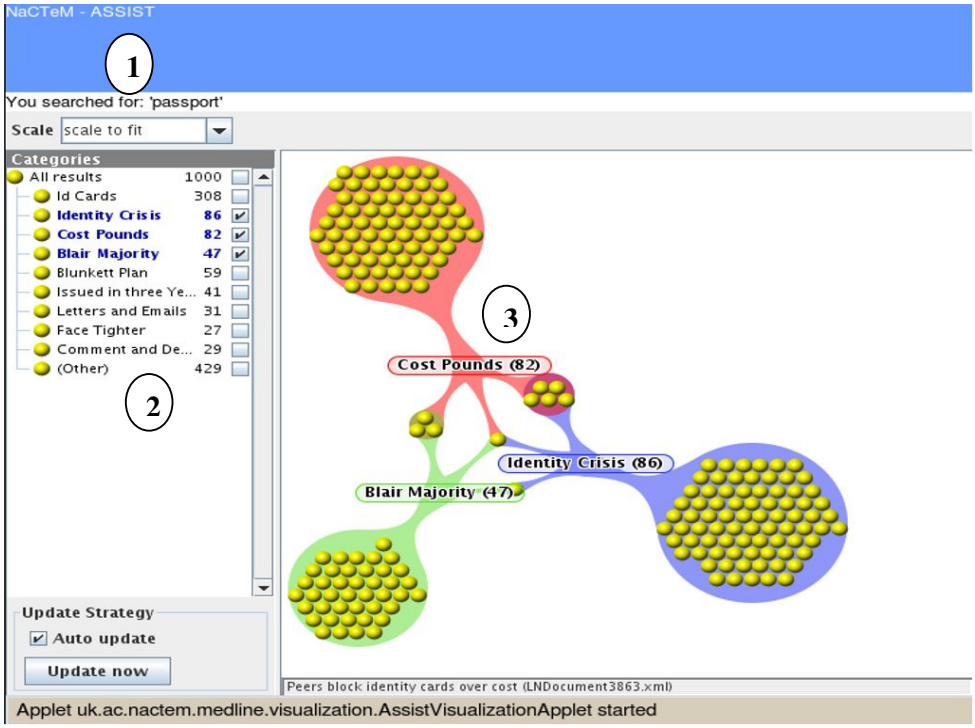

Illustration 4: A graphical representation of the clusters

The last part, numbered 2, informs the user of all the clusters computed for his query and the number of documents they contain. When the box in front of a cluster's title is clicked, the cluster is added to the graphical representation. In our illustration three clusters titled *'Identity Crisis', 'Cost Pounds'* and *'Blair Majority'* have been selected.

2.5 Document visualization interface

When the user clicks on the title of a document from the search result interface, the full document is displayed in the document visualization interface.

The interface presenting the full document is composed of three parts. The first part (number 1 in Illustration 5) displays the full document. The first line identifies the document with its URL (in the illustration, the document LNDocument1135.xml). The document is displayed with its title in bold, when available. In the content of the document, different pieces of information will be highlighted. In the current version only the terms have been implemented and highlighted in pink (*e.g.* 'biometric passports' in our illustration). The named entities[6] and precise concept representation with sets of related terms[7] will be added in future versions.

The part numbered 3 in the illustration titled '*Related Topics*' lists all the terms contained in the document. Each term in the list is a hyperlink. By clicking on a hyperlink, the user sends a new

[6] The named entities are computed with the Text Mining tool BaLIE [Nadeau, 07].

[7] For more details about this representation of a concept see http://www.csi.ucd.ie/staff/jcarthy/home/Lex.html

query composed of the highlighted term, and the cluster visualization interface is displayed with the new results. Three Hyperlinks *'Low', 'Medium'* and *'High'* are displayed on the top of the terms list. By clicking on these hyperlinks, the user changes the value of the threshold for the filter and selects different sets of terms. For example, by clicking on the hyperlink 'Low', the value of the threshold is lowered, terms with lower scores are selected, and consequently the list is longer.

The part numbered 2 in the illustration titled '*Related Document*' proposes to the user various documents related to the document displayed in part 1. The related documents are identified by their titles, and a hyperlink allows the user to display one of these documents in part 1 of the interface. The related documents are chosen according to the number of meaningful words they have in common. For an introduction to the similarity between documents used by our search engine see [Gospodnetic & Hatcher, 05].

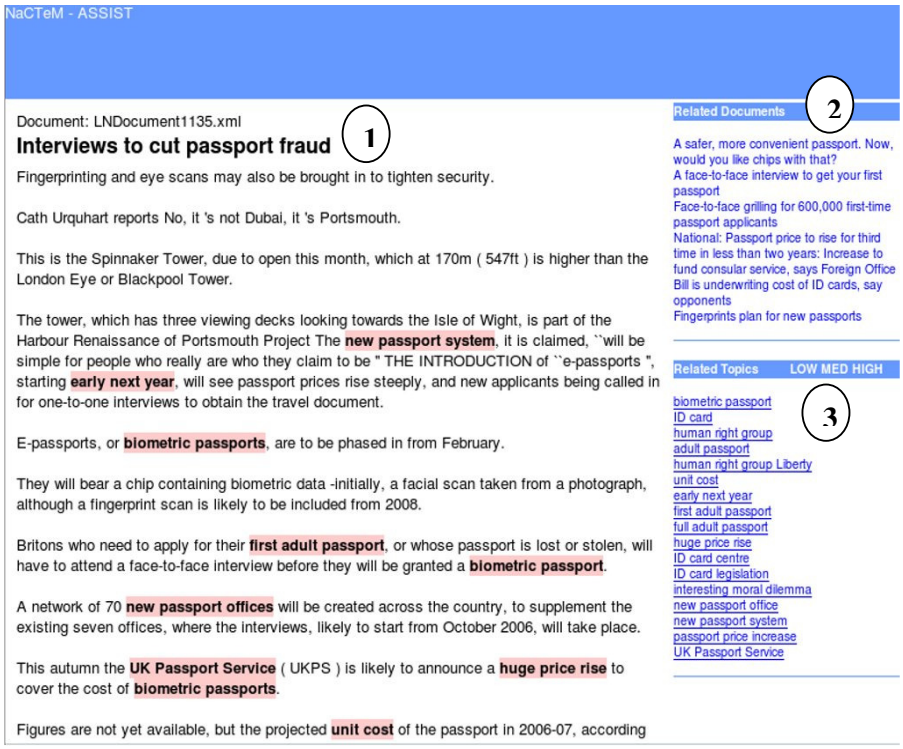

Illustration 5: Document visualization interface

References

O. Gospodnetic & E. Hatcher, *Lucene in Action*, Manning Publications, 2005.

S. Osinski, *An algorithm for clustering of web search results*, Master Thesis, Poznan University, Poland, 2003.

K. Frantzi, S. Ananiadou and H. Mima, *Automatic recognition of multi-word terms,* International Journal of *Digital Libraries* 3(2), pp.117-132. 2000.

D. Nadeau, *Semi-Supervised Named Entity Recognition: Learning to Recognize 100 Entity Types with Little Supervision.* Thesis of the University of Ottawa, Canada. 2007