ASSIST project - internal report ASSIST-D1, Davy Weissenbacher, Yutaka Sasaki, Tingting Mu, Brian Rea, Sophia Ananiadou, November 2008

# ASSIST: A Specialized Search Engine for the social sciences using text mining

## *1. System architecture*

This document reports technical components of the prototype for the ASSIST search-engine[1]. A tutorial explaining how to use this web-based system is available as a separate document, the internal report ASSIST-D3.

### 1.1 Focusing on Stakeholders needs

To design an efficient text mining (TM) tool meeting the users' expectations, it is crucial to have close interaction with them in the earlier stages of the conception. To ensure this, we have adapted to the social science domain an existing search engine developed during the ASSERT project[2].

The ASSERT search engine is a modular platform allowing us to plug in advance text mining tools. Extending the ASSERT search engine with existing TM tools we intend to get a simple and complete prototype to elicit user requirements in a collaborative way. Based on the users feedback received from their prototype evaluation we can continuously update and improve the embedded TM tools to obtain the final ASSIST search-engine.

In the next section we present the architecture of the prototype used for the first evaluation of our stakeholders. In section 2 we describe the TM tools integrated in the prototype. At this stage of the project the prototype does not integrate all the TM tools planned, we restrict the discussion for the integrated TM tools. In the last section we discuss how the TM tools can enhance the search for relevant document and the current limitations of the components.

### 1.2 Prototype overview

#### 1.2.1 Indexing the documents

Based on the Lucene library[3], our search engine first processes the corpus to index all the documents. The general architecture of our indexer is shown in the Figure 1.

[1] For references about the project see http://www.nactem.ac.uk/assist/

[2] For references about the project see http://www.nactem.ac.uk/assert/

[3] The Lucene library is available at http://lucene.apache.org/java/docs/

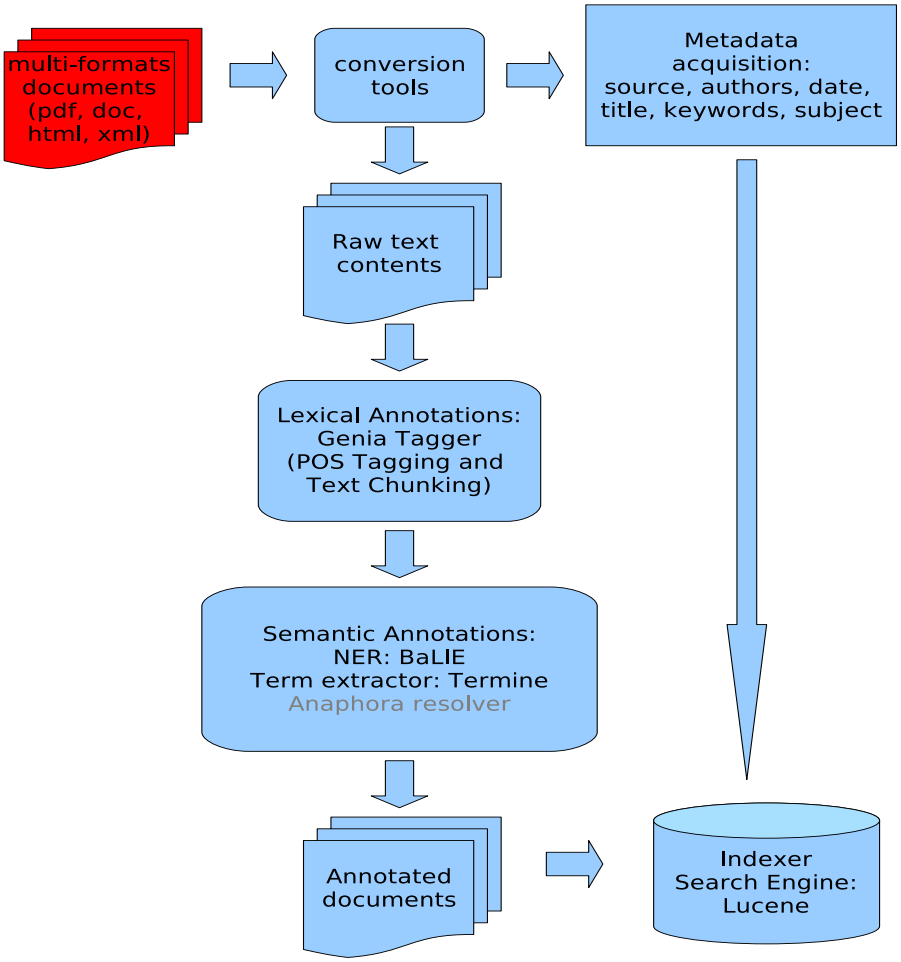


**Figure 1: ASSIST indexing pipeline**

### *1.2.1.1 Documents Conversion*

The first stage of the indexing process is the conversion of input documents to suitable data formats. Our corpus is composed of many formats documents, namely PDF, HTML, XML documents and Word Microsoft Office documents. These formats cannot be indexed directly by Lucene and have to be converted in text only representations. The conversion of a particular format is addressed by a specific converter. These converters are detailed in the section *3.1 Content and Metadata Acquisition*.

During the conversion the system produces two outputs. The first output is the metadata specifying the origin of a document (e.g. the authors or the date of publication of a document). These metadata depend on the format of the document and are integrated in the document. The system extracts from the document these pieces of information and indexes them separately allowing research on them during the latter stage. The second output is the content of the document in raw text format. Following the removal of the figures, tables and images the content of document is presented to the TM tools in order to be enriched with lexical and semantic annotations[4].

[4] The concept of annotation which we refer to in this report is formally defined in the section 3.1 Content and Metadata

#### *1.2.1.2 Documents Annotation*

The first process of the document is the lexical annotations. This lexical annotation consisting in a POS tagging and a text chunking is performed by the Genia tagger. the Genia tagger is used in the original ASSERT search engine on which the ASSIST search engine is based on. References and performances of this TM tool can be found from http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/tagger/.

The content of the document annotated with lexical information is needed to perform the next process, the semantic annotation. Different TM tools, using the lexical annotation as input, process the document to add their own annotations. The current version of the prototype integrates the named entity recognizer called BaLIE[5] (presented in the section *2.1 Named Entity Recognizer*) and the term extractor Termine (presented in the section *2.2 Terms Extractors*). The future version will integrate an anaphora resolution system. BaLIE and Termine are applied independently on the document. The document annotated is then inserted in the index for future searches.

## 1.3 Searching the documents

The index built up is used by the search engine to retrieve pertinent documents according to a user query.

---

Acquisition

[5] BaLIE is available at http://balie.sourceforge.net/

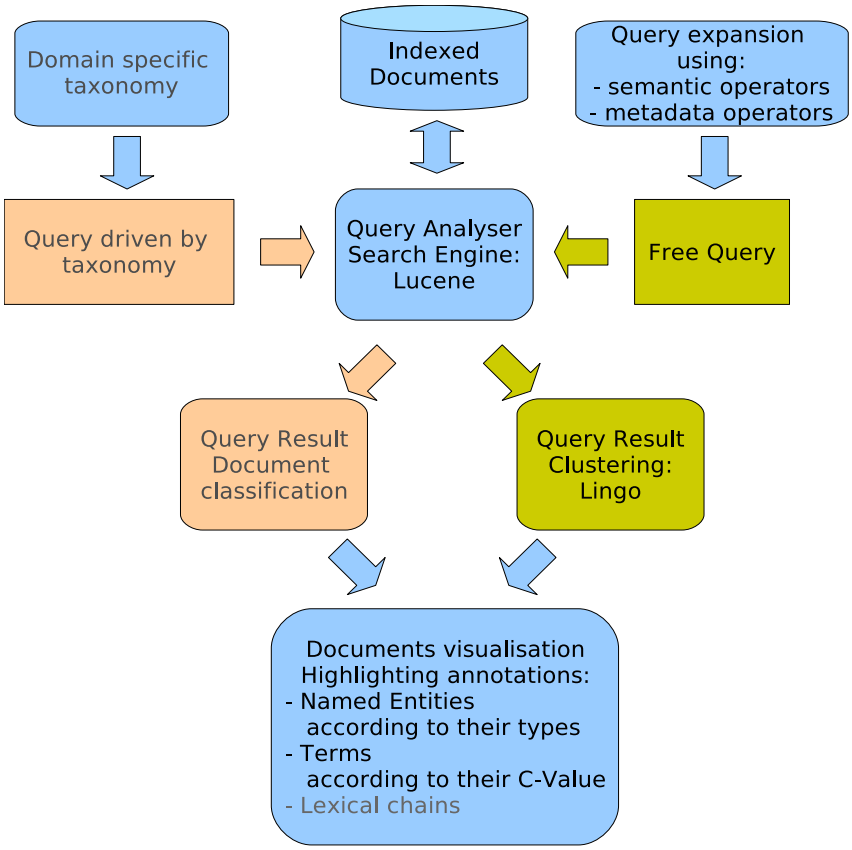


**Figure 2: ASSIST searching pipeline**

The current version of the prototype proposes to search documents according to a classical Boolean query using Lucene predefined wild-card characters. This query can be extended with a set the semantic operators thanks to the semantic annotations or with specific operators to query the metadata associated to the documents. The operators available for this prototype are described in sections *3.4 Expanding the user query*. A future version of the prototype will integrate a domain specific taxonomy provided by our stakeholder. Driven by the hierarchy of the concepts in the taxonomy the user gets the possibility to browse the corpus according to the concepts he is interesting in.

Depending on the type of the query, a free query or a query driven by the taxonomy, the query result is processed differently. For the query driven by the taxonomy the search engine has to retrieve all the documents associated with the concept of the taxonomy queried (even if this concept is not explicitly mentioned in the document). This problem is a document classification problem. For a free query the search engine retrieves a list of documents relevant for the query. If the list is too long the result is not informative and the documents must be clustered. This problem is an unsupervised clustering problem. We discuss both problems in the section *3.5 Query results clustering and classification*.

When a document is chosen from the query result, it is displayed with various annotations highlighted. These annotations give fast access to the content of the document focusing the reader on the mains pieces of information of the document. The annotations available in this prototype are named entities and terms.

## *2 Presentation of individual Text Mining components*

In this section we describe the text mining tools integrated in the prototype. After a short presentation of the annotation added by the tools we report the evaluation published by the author(s). We stress the limits of the external tools and suggest possible improvements or tools to replace them.

## 2.1 Named Entity Recognizer

### 2.1.1 Named Entity Definition

Traditionally, Named Entities (NEs) are a noun phrase used as a rigid designator [Kripke, 82] to denote an existing object in a real or an imaginary world.

Named Entity recognition is an important step in text mining. To understand the meaning of sentences and extract useful information, we need a fundamental tool that can correctly detect the meaning of words in the sentences. This is quite easy for humans but it is not the case for software systems. For example, person names are highly variable and it would be very difficult to build a dictionary that contains all the person names in the world. New names are created everyday. It is also problematic that many person names are used for general meanings as well as other named entities. For example, Washington can be used for a person name but it could also be a location name. Company and shop/restaurant names are more confusing as they can be any kind of names, such as "river" or "X".

First NEs recognizers focused on three main categories of named entities: (1) person names (e.g. George W. Bush, Mr Chirac, Harry Potter, etc.), (2) organization names (e.g. Microsoft, ONU, Ford, etc.), and (3) numerical expressions, i.e., date, money, and percentage. These three main categories of NEs appear too general to be useful. Consequently the categories of the named entities have been extended [Sekine & Nobata, 04]. Creating a large hierarchy of named entities, this work specifies new subcategories of existing entities and adds new types of entities. For example the category of names still contains the names of people, but this category is subdivided into subcategories to denote the names of fictional characters, nicknames, etc. At the same time, the category of names is extended with new types of NEs like names of products (e.g. vehicle, food, printing, etc.). The extension of the hierarchy of NEs increases the probability to recognize the main discourse objects in a document and refining the categories of the hierarchy provides to the system precise types for the NEs which are precious semantic annotations.

### 2.1.2 Named Entities Recognition

Research into Named Entities recognition is centred around three approaches: dictionary-based, rule-based, and machine learning-based approaches [Ananiadou & McNaught, 06].

To automatically extract predefined NEs in a particular corpus, the first strategies took by the NE recognizers were the dictionary-based and rule-based approaches. For example, consider the rule

'IF the sequence contains a title (e.g. Dr., Mr., etc.) followed by a word A which is not within the general language dictionary, THEN the word A is a Named Entity'. This simple rule extracts 'Clinton' as a named entity from the sentence 'While Mr. Clinton has used his foundation to champion […]'. These rules are very precise, but their performance fall down when they were applied on another domain for which they have been written [Poibeau, 03] (dictionaries used in the rules are not exhaustive or not available or the rules don't match a sequence because of unforeseen linguistic variations, etc.).

To avoid this drop in performances, the NE recognizers based on supervised machine learning tend to take the rule-based NER places. The machine learning-based systems learn automatically the extraction rules according to the domain of the corpus on which the NE recognizer is supposed to be applied. A supervised machine learning algorithm exploits positive and negative examples of NEs given through a human-annotated corpus to learn the internal and external contexts of the NEs. Knowing the contexts, the machine uses them to formulate the rules for extraction. The learning stage automatically adapts the NE recognizer to the domain, changing the rules it uses. Unfortunately, to learn the rules the system needs numerous examples which are rarely available and the annotation of the examples is a time and effort consuming task. For this reasons the community tries semi or unsupervised learning method to annotate NE [Nadeau & Sekine 07]. The next section presents BaLIE a recent semi-supervised machine learning-based NE recognizer.

### 2.1.3 BaLIE a semi-supervised machine learning-based NE recognizer

One of the current challenges for the NE recognition is to reduce the number of examples needed by the machine learning system. One of the possible strategies is to use a NE recognizer based on semi-supervised learning. The main idea of this strategy is to exploit the natural redundancy in the huge amount of unlabelled data to automatically extend a small seed of examples.

We have selected an open source java program under GNU GPL licence called BaLIE [Nadeau *et al.*, 06]. This system is a complete NE recognizer using semi-supervised machine learning and the Yahoo search engine to extend its gazetteers. Given that the named entities of interest are unknown and will probably change in accordance with the work of our stakeholders, we have favoured the semi-supervised learning strategy to improve the adaptability of our NE recognizer and reduce the cost of its adaptation.

The BaLIE system is composed of two modules. The first module, the NE Extraction module, generates from web pages a list of NE, called gazetteers. However the NEs obtained are ambiguous and cannot be applied immediately to the corpus of interest without a drop of the Precision[6]. The

[6] The performance of a NE recognizer are usually measured in term of recall, precision and the overall F-score:

Precision = number of NE identified correctly/( number of NE identified correctly + number of string mistaken claimed to be entities)

Recall = number of NE identified correctly/( number of NE identified correctly + number of NE not identified)

F-Score = 2*Recall*Precision/(Recall + Precision)

second module, NE disambiguation, implements different heuristics proposed in previous state-of-the-art works to filter out highly ambiguous NEs recognized in a document context. We explain in details each module and the performances obtain by this system on the MUC-7 corpus.

#### *2.1.3.1 NE Extraction*

To acquire a list of NEs the module starts with a seed of NEs of interest given by the user. The module selects *k* NEs from the seed and submits a query $(NE_1 \ AND \ldots \ AND \ NE_k)$ to a search engine. The idea is a web page which contains *k* known NEs of the same type should contain, at least, a new unknown NE of this type. For the author experiments *k* is fixed to 4 after human observations on a dry run. The extraction of the unknown NEs from the query results is formulated as a classification problem: discriminate the HTML nodes which contain a NE from others nodes. The author realises this classification using linguistic features to describe the sequence in a node as well as features to describe the node in the HTML structures, the details for the problem resolution are published in [Nadeau, 05].

When the new NEs have been extracted, the module chooses *k* different NEs from the obtained seed. The algorithm chooses the most reliable NE from the seed which are the NEs appearing in several documents. A new query is formulated and the process iterates.

#### *2.1.3.2 NE Disambiguation*

This module has to annotate the NEs in a document. Processing the sequences of the document, the module annotates the sequences matching a NE in a gazetteer built by the NE extraction module. However this module performed badly, the precision drops due cause to ambiguous NEs. The author counts 3 types of ambiguities and proposes different heuristics to resolve the ambiguity using the internal or external contexts of a document sequence matched in a gazetteer. If the sequence is still ambiguous after the rules application, it is not annotated as a NE by the module.

- Entity-Noun ambiguity: This ambiguity rises when a NE is a noun homograph (e.g. jobs the plural noun for job and Jobs the family name). A rule proposed to disambiguate these NEs is, for example, by default the sequence matched in the document is a NE except if this sequence is not capitalized, in this case the sequence is a common noun and is not be annotated.
- Entity boundary detection: The detection of the exact boundary of a NE could also be a problem. Consider the example given by the author, if the module matches the sequence Boston in a sentence speaking about the baseball team Boston White Sox, it has not to annotate the sequence Boston as a city but annotate the full sequence as a sport team. The author proposes different rules to annotate the longest sequence supposed to be a NE.
- Entity-Entity ambiguity: The last ambiguity occurred when a string denoting a NE is found in two or more gazetteers. The module has to find the correct type for the NE as for France which can be the country or a last name. An algorithm has been proposed in [Nadeau *et al.*,

06] to find clear evidence in the external contexts of the NEs for their types.

### *2.1.3.3 BaLIE Evaluation*

To evaluate the interest of the semi-supervised learning based methods the author compares the performances of two systems in [Nadeau *et al.*, 06]. The first system is the BaLIE system and the second is the [Mitkheev *et al.*, 99] system, a supervised machine learning based system which was amongst the higher scoring in the evaluation campaign MUC-7[7]. The table summarizes the results performed by three different NE recognizers on the MUC-7 corpus. The systems have to recognize three types of NEs: the organisation, the nouns of persons and the names of locations. The first column of the *Table 1* presents the scores obtained by the [Mitkheev *et al.*, 99] supervised learning based system corpus. If the precision claims for this system is good, it suffers from a weak recall, except for the location. On the opposite, when BaLIE is used without any disambiguation rules (*i.e.* in this case the gazetteers learned by the NE Extraction module are applied in the test corpus and all sequences matches are annotated as NEs) the recall is very high but, due to the ambiguous NEs, the precision is low, as we can see in the second column. Filtering out the ambiguous NEs, the complete system achieves better F-scores than the [Mitkheev *et al.*, 99] system (third column). These results are a notable contribution: with less human intervention to annotate the data a semi-supervised learning based system gets comparable performances as supervised learning based system.

**Table 1: Performance of supervised and semi-supervised machine learning based systems**

| | [Mitkheev *et al.*, 99] system | | | BaLIE without Disambiguation rules | | | BaLIE with Disambiguation rules | | |
|---|---|---|---|---|---|---|---|---|---|
| | R | P | F | R | P | F | R | P | F |
| Organisation | 50 | 72 | 59 | 70 | 52 | 60 | 71 | 75 | 73 |
| Person | 47 | 85 | 61 | 59 | 20 | 30 | 83 | 71 | 77 |
| Location | 86 | 90 | 88 | 83 | 31 | 45 | 80 | 77 | 78 |

### *2.1.3.4 BaLIE Advantages and Limitations*

We claim that the use of this strategy is perfectly justified for this project. At this stage of the project our stakeholders are unable to define the exhaustive list of the types of NE they are interested in, but, based on the evaluation of the first prototype, they know that common types of organisation, person and location are too general and not cover all the NEs of their interest. A system which allows us to build up quickly huge gazetteers for specific types of NEs with a minimum human intervention is necessary.

[7] Description of the evaluation campaign can be found in http://www-nlpir.nist.gov/related_projects/muc/proceedings/muc_7_toc.html

Once the list of the types of NEs will be defined and agreed by all the stakeholders of this project, we will proceed to the extraction of the NEs and carry on with a complete evaluation of the BaLIE performance for these new types of NEs on our corpus.

The comparison between the performances of BaLIE with and without the disambiguation rules shows the limitation of this strategy. The performance of the system depends mainly of the disambiguation rules used. The author confesses that these disambiguation rules are simple heuristics. Only some experimental results can justify their use. Nothing guarantees correct performances from them when they are applied for different types of NE[8]. To anticipate this limitation we are considering, as a future work, a hybrid approach to take advantages from both semi-supervised and supervised learning strategies. We describe this approach in the next section.

### 2.1.4 Future work, a dictionary-based statistical NER

To overcome the usual NER pitfalls, we are investigating a possibility to take a hybrid approach combining dictionary-based and machine learning approaches, which we call dictionary-based statistical NER approach. The basic idea is to use existing NE dictionaries to cover known names and revise the initial results with statistical NER trained on an annotated corpus. As addressed before creating training data is costly and time consuming, we are considering to utilize BaLIE's outputs for NE training data. The reason why we are considering this approach is that the dictionary-base statistical approach was quite successful in protein name recognition tasks [Sasaki et al., 2008].

Figure 3 shows the block diagram of dictionary-based statistical NER. Raw text is analyzed by a POS/PROTEIN tagger based on a CRF tagging model and dictionary, and then converted into token sequences. Strings in the text that match with protein names in the dictionary will be tagged as NN-PROTEIN depending on the context around the protein names. Since it is not realistic to enumerate all protein names in the dictionary, due to their high variability of form, instead previously unseen forms are predicted to be protein names by statistical sequential labelling. Finally, protein names are identified from the POS/PROTEIN tagged token sequences via a CRF labelling model.

8 A non-free version YooName extends BaLIE with new types of Nes recognized, NEs filter out with adding disambiguation rules: http://www.yooname.com/

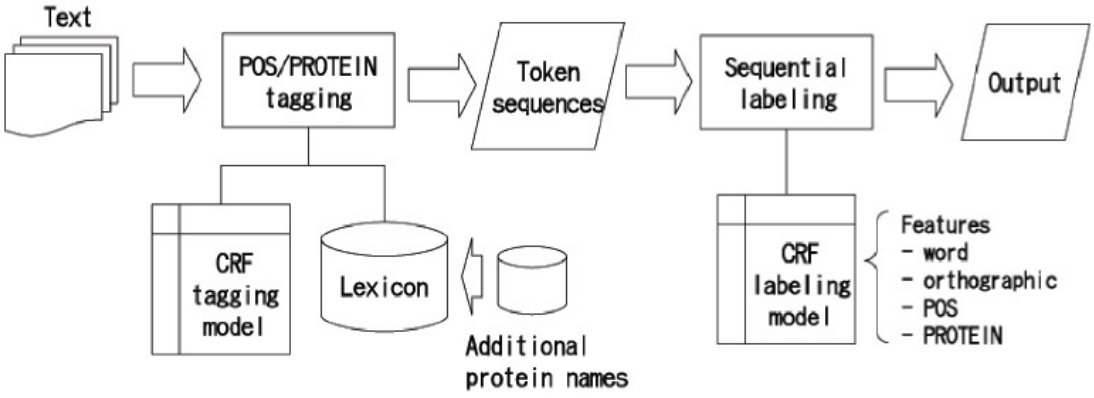


**Figure 3: Block diagram of dictionary-based statistical NE recognizer**

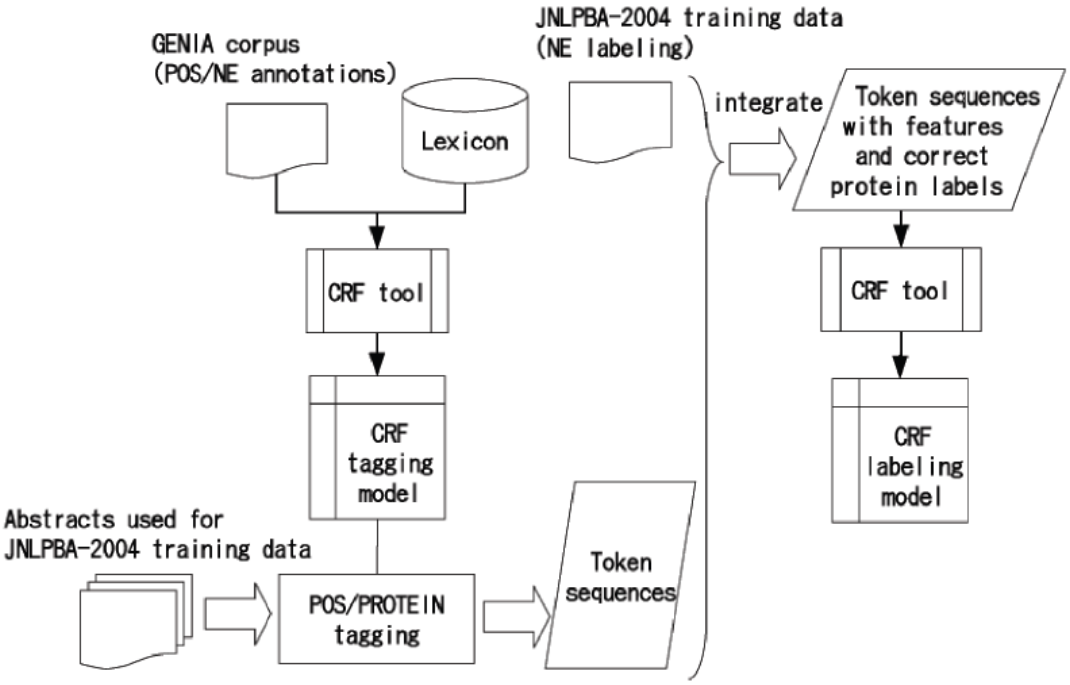


**Figure 4: Block diagram of training of dictionary-based statistical NE recognizer**

Figure 4 shows the block diagram of the training procedure for both POS/PROTEIN tagging and sequential labelling. The tagging model is created using the Genia corpus (version 3.02) and a dictionary. Using the tagging model, MEDLINE abstracts used for the JNLPBA-2004 training data set are then POS/PROTEIN-tagged. The output token sequences over these abstracts are then integrated with the correct protein labels of the JNLPBA-2004 training data. This process results in the preparation of token sequences with features and correct protein labels. A CRF labelling model is finally generated by applying a CRF tool to these decorated token sequences.

The NER tool was trained with Conditional Random Fields (CRFs) [McCallum et al., 00] on the training set of the JNLPBA-2004 data set [Kim et al., 04]. The training data set used in the JNLPBA-2004 shared task is a set of tokenized sentences with manually annotated term class labels. The sentences are taken from the Genia corpus (version 3.02) [Kim et al., 03] in which

2,000 abstracts were manually annotated by a biologist. In the JNLPBA-2004 shared task, performance in extracting five named entity classes, i.e. protein, DNA, RNA, cell line and cell type classes, was evaluated.

The test data set used in JNLPBA-2004 is a set of tokenized sentences extracted from 404 separately collected MEDLINE abstracts, where the term class labels were manually assigned, in accordance with the annotation specification of the Genia corpus.

Statistical sequential labelling was employed to improve the coverage of protein name recognition and to remove false positives as well.

Following the data format of the JNLPBA-2004 training set, our training and test data use the IOB2 labels [Tjong Kim Sang & Veenstra, 1999], which are ``B-protein" for the first token of the target sequence, “I-protein” for each remaining token in the target sequence, and ``O" for other tokens.

For example, “Activation of the IL 2 precursor provides” is analyzed by the POS tagger as follows:

```
Activation      NN
of              IN
the             DT
IL              NN
2               CD
precursor       NN
provides        VVZ
```

The tagger output is given IOB2 labels as follows:

```
Activation      NN    O
of              IN    O
the             DT    O
IL              NN    B-protein
2               CD    I-protein
precursor       NN    I-protein
provides        VVZ   O
```

We applied CRFs to predicting the IOB2 labels. The following features were used in our experiments.

- word feature
    - orthographic features, the first letter and the last four letters of the word form, in which capital letters in a word are normalized to “A”, lower case letters are normalized to “a”, and digits are replaced by “0”. For example, the word form “IL-2” is normalised to “AA-0”.
    - postfixes, the last two and four letters
- POS feature

- The window size was set to ±2 of the current token.

Results are expressed according to recall (R), precision (P), and F-measure (F), which here measure how accurately the various experiments determined the left boundary (Left), the right boundary (Right), and both boundaries (Full) of protein names.

Table 2 shows the evaluation results. The baseline for sequential labelling was that of the prediction performance when using only word features where no orthographic and POS features were used. The F-score of the baseline labelling method was 66.62. The F-score of the model trained with all the features improved from 66.62 to 73.78, which is the second best score for protein name recognition among research reported using the standard JNLPBA-2004 data set.

**Table 2: NE recognizer performance**

| | | R | P | F |
|---|---|---|---|---|
| Sequential Labeling | Full | 79.85 | 68.58 | 73.78 |
| | Left | 84.82 | 72.85 | 78.38 |
| | Right | 86.60 | 74.37 | 80.02 |

We believe that combining BaLIE and the dictionary-based statistical NER approach will be one of the possibilities that we can achieve high performance with less corpus annotation efforts.

## 2.2 Terms Extractors

### 2.2.1 Term Definition

A term can be defined as a linguistic form of a concept admitted by a community of the domain [Frantzi et *al.*, 00]. For example 'm*usic education research*' or 'information and communication technology' are terms of the education research domain. In our examples the terms are noun phrases, but they can be verb phrases, adjective phrases, *etc.* However, these latter phrases are not available in our version.

A noun phrase term is not necessary the longest linguistic form where it can appear. Consider the ophthalmology term given by [Frantzi et *al.*, 00], '*soft contact lens*'. This term contains as a substring another term '*contact lens*'. This second term is a nested term. Note that some nested terms can appear by themselves in the corpus, *i.e.,* not as a substring of a longer term but as an individual entity (e.g. '*A contact lens (also known simply as a contact)*'). Other nested terms always appear within longer terms; for example 'real time' in the computer science domain appears in '*real time clock', 'real time output', 'real time systems', etc.* Even if this term is always nested, it is a term because it shows an independence from the longer terms where it appears.

### 2.2.2 Term Extraction with Termine

To recognize and extract the terms automatically, we have integrated the NaCTeM's tool *Termine*[9]. This component is domain independent and exploits linguistic information as well as statistical information to identify the terms and the nested terms in a corpus. Two stages can be distinguished in the algorithm of this component.

- Linguistic filters (Stage 1): The processing of the corpus starts with a POS tagger to annotate the syntactic classes of each word in a sentence (e.g., noun, verb, adjective, etc.). Then syntactic patterns are applied to extract the noun phrases of the corpus. These patterns are abstract descriptions of various noun phrases forms:
    - Noun+ Noun: at least one noun followed by another noun, e.g. '*computer game software*', '*network device*', *etc.*
    - (Adj|Noun)+ Noun: at least one noun or adjective followed by another noun, *e.g.* '*optical mouse*', '*soft contact lens', etc.*
    - …

A stopword dictionary is applied to the resulting list of noun phrases. This operation removes from the list all noun phrases containing one word of the dictionary, e.g. *great, good, etc.* Such noun phrases are probably not terms.

- Statistically-based extraction (Stage 2): The list of the noun phrases is the list of candidate terms. The frequency of each candidate term is computed on the corpus. A filter based on a given frequency threshold is applied to remove candidates which have too few occurrences in the corpus. Four forms of statistical information are required to decide which remaining candidates are terms:
    - the total frequency of occurrence of the candidate string in the corpus
    - the frequency of the candidate string as part of other longer candidate terms
    - the number of these longer candidate terms
    - the length of the candidate string (in number of words)

These pieces of information are involved in the calculation of an individual score for each candidate term. This score is called the C-value. The following sentences give an intuitive explanation of the equation.

If the candidate term is never found as a nested term in the corpus, then its frequency is a good indicator. The length of this candidate causes the frequency indicator to favour the longer candidate terms which, with a weak probability to appear, are certainly terms. Otherwise, the candidate term appears within a longer candidate term. This position is a negative feature in concluding that the candidate term is a term. To correct it, other statistical information is used in the equation: the frequency of this candidate term appearing by itself (to catch the nested terms which appear also as individual entities) and the number of terms where the candidate term is a substring (to catch the nested terms which always appear within longer terms). After the C-value is calculated for each

[9] A web demonstrator and bibliographical references can be found at http://www.nactem.ac.uk/software/termine/

candidate term, the list of candidate terms is ordered and all the candidates above a given C-Value threshold are designed as terms for the domain of the corpus.

### 2.2.3 Termine Evaluation

It's difficult to evaluate the list of terms quantitatively even for a domain specialist. A term is defined as a consensual linguistic representation of a concept by the community. A domain specialist clearly knows that some candidates are or are not terms of the domain. For others, because these candidates are obsolete or not consensual, the domain specialist is not always able to decide if a phrase is a term or not. For this reason one measure proposed to score the performance is the relative Precision and Recall. Termine has already been evaluated in detailed on a medical corpus and we refer to the [Frantzi et *al.*, 00] for the overall results.

The first version of our prototype gives us the opportunity to evaluate the precision of Termine on our stakeholders' corpus (both are described in the internal report ASSIST-D3). Only the precision measure will be investigate for this project. Recall evaluation is a time-consuming task and is not essential for this application. The search engine returns a list of terms to synthesize the main topic of the documents. The precision of the selected terms is then important whereas a term can be missing as long as the topic is capture with the other extracted terms.

The protocol to evaluate the precision of the terms is defined as follows. Given a query result a list of documents is returned. Each document is associated with a list of terms which are supposed to represent the main topics of the document. Our stakeholder, a domain expert, has to evaluate the quality of the terms. For the corpus of the education research domain, the terminology is well formalized and we expect an objective evaluation of the precision from our expert. For the mass-media corpus, the terminology is not defined. So we have asked our expert to measure the relevance for a term for describing the topic of the document. This measure can be defined as a subjective evaluation of the precision of terms. The evaluation is still in progress. We will produce the results in the next technical report.

## *3 TM components to improve the Information Retrieval task*

## 3.1 Content and Metadata Acquisition

### 3.1.1 Formats Conversion

Our prototype is able to process different formats of documents, namely *'XML', 'HTML', 'PDF'* and Microsoft Office Word formats. The LexisNexis corpus created for the NCESS search engine has been reformatted in an XML format from its original structured in rich text format but the XML format facilitates the automatic processing of the documents. The EPPI corpus comes from various educational web sites and is composed of 'HTML', 'PDF' and Microsoft Office Word documents. We have used existing and publicly available converters to extract the textual contents of

documents as raw text. *PDFbox*[10] has been selected to process 'PDF' documents. *POI*[11] has been selected for is convenient representation of the Microsoft Word documents. *Jtidy*[12] has been selected because it cleans ill-formed *HTML* documents, and it allows us to access all the *HTML* tags and textual contents conveniently.

### 3.1.2 Documents Contents Extraction

The conversion of multi-format documents to raw text is not a simple task. Due to the limitations of our converters or the diversity of the formats of the documents, the content of the documents are not perfectly extracted. We describe here the errors in the contents extract and the solution implemented to reduce their occurrences.

After processing HTML documents with the parser, the logical structure of the file is lost. The title, subtitles, lists and references are flattened into a linear text. The consequence is the creation of ungrammatical sentences like "*Key findings Use of formal tests Over 95 per cent of schools in the survey sample use the QCA optional tests in English and mathematics.*" To solve this problem we have adapted the output of the Jtidy parser to preserve the important parts of the logical structure of the HTML documents using the HTML tags.

With HTML and Microsoft Word documents, we have seen the appearance of unexpected non-alphanumeric characters and words which are not in the original content of the document (*e.g.* the index, hyperlinks, footnotes...). To fix this problem we added a cleaning stage which deals with each type of unexpected character and word independently. It should be noted that the proposed solutions are not perfect because of the diversity of the documents. Ungrammatical sentences and unexpected characters remain in indexed documents.

### 3.1.3 Metadata Extraction

#### *3.1.3.1 Task Definition*

The concept of annotation has been defined in the technical report TIPSTER [Grishman, 1997]. An annotation is a predefined property associated to a continuous or discontinuous sequence of a document (*e.g.* a POS tag or a NE). Some annotations are associated to the document itself. This type of annotations is usually called metadata of the document. The metadata refer to the information located before or after the full content of the document:

- the title of the document
- the author(s) of the document, their affiliations, e-mails, address, phone
- the keywords describing the topic of the document

10 Website: http://www.pdfbox.org/

11 Website: http://poi.apache.org/hwpf/index.html

12 Website: http://jtidy.sourceforge.net/

- summarize of the content
- bibliographical references
- ...

The automatic extraction of metadata is a new challenge for the TM community. To maintain the coherence of their database and to offer powerful search operators, the digital libraries extracted manually the metadata of their documents. With the increase in number of digital documents this manual extraction is no longer possible and automatic solutions based on Text Mining methods have to be designed. The first works implement Information Extraction techniques to extract the metadata [Han et al., 05], [Ivanyukovich & Marchese, 06]. Several factors can make the extraction difficult and reduce or increase the performances:

- the format of the documents: pdf, html, Word documents, Excel documents, *etc.*
- the type of document: research publication, news papers, poster, thesis, *etc.*
- the type of metadata extracted: authors, bibliographic references, keywords, *etc.*

According to these parameters, important variations in the performances are observed [Han et al., 05] published performances from 50% to 95% in f-score depending on the type of metadata.

### *3.1.3.2 Metadata Extraction for our Corpora*

The corpus created for our stakeholder NCeSS has been built up from the digital library Lexis Nexis. The corpus is composed of 4889 articles of different news papers. The documents are available in RTF. The logical structure of these documents is quiet regular: the source of the article is stated in the first line, then in order of display, the date of publication, the title, various metadata explicitly mentioned, the content, the author(s) and other various of metadata. We give an example of the document in the Table 3. Making use of this regularity, we have formatted the corpus into the XML format, which is more convenient for further processing. Exceptions in the logical structure of some documents lead the program to leave some occurrences of metadata empty, but the extraction is well performed on this corpus. The number of missing occurrences for each type of metadata is given in the Table 4.

**Table 3: Example of Document in RTF format**

| |
|---|
| The Sun (England) |
| May 21, 2008 Wednesday |
| Hijack job's plane crazy |
| SECTION: LETTER |
| LENGTH: 62 words |
| THE Government want to convince us to back their loony ID card scheme by issuing them first to people working at airports - to stop the "wrong people" working there and endangering travellers. |
| If the authorities are happy to let proven hijackers like Nazamuddin Mohamiddy work at Heathrow, just who do they consider to be the "wrong people"? |

GEORGE EDWARDS

Rawcliffe, E Yorks

LOAD-DATE: May 21, 2008

LANGUAGE: ENGLISH

PUBLICATION-TYPE: Newspaper



**Table 4: Metadata extracted in the Lexis Nexis corpus**

| Type of Metadata | Missing Occurrences |
|---|---|
| Source | 1 |
| Authors | 1277<br>Most of these documents are letters send to the news papers and written by anonymous. |
| Date | 0 |
| Title | 35 |

The corpus provided by our stakeholder EPPI is more problematic. The corpus is composed of 1300 documents extracted from various educational web sites. The documents are in PDF, HTML and Microsoft Office Word format. Each format proposes different metadata with different logical structure making the automatic extraction very difficult.

- PDF documents: the metadata can be found in a data cartridge, but this cartridge is often left empty or filled with inconsistent information (*e.g.* in the author field, we can find the value '*user*').
- Word documents: functions are designed to return metadata associated with the document. Unfortunately the Word documents format is not available and these functions cannot isolate the metadata and returns unexpected information (*e.g.* a function does not return the title but the text in the logo placed before the title in the document).
- HTML documents: in this format some metadata are clearly mentioned with meta-tags and can be extracted easily. However, some metadata are not tagged and are inserted in the content of the document (*e.g.* there is no meta-tag to mark the authors, and they are not emphasized with an HTML tag in the documents).

The application of information extraction techniques for this purpose is beyond the scope of this project. We have implemented simple solutions to recognize the main metadata but the results is clearly insufficient. In accordance with our stakeholder we have delayed the resolution of this problem for future work in order to focus on the integration of the TM tools in the prototype.

## 3.4 Expanding the user query

The main approach for the next generation of search engines is to complement a search engine with TM tools to help the user in his/her search. The documents are enriched with the annotations produced by TM tools before being indexed. This enrichment makes new operators available to query the search engine and refine the query results.

The operators to query the metadata attached to documents are the first of TM based operators. As discussed in the section 3.1.3 Metadata Extraction, the metadata are semantic annotations about the origin of documents that can be used to select a specific set of documents, for instance all the documents written by an author during a certain period.

The internal report ASSIST-D3 details all the metadata operators available for this prototype. We summarize the operators available according to the corpus:

- NCeSS corpus:
    - *authors:* to search for a specific author or co-authors of a document
    - *date:* to search for a date of publication of a document
    - *source:* to search for documents published by particular news papers
    - *title:* to search for words appearing in the title
- EPPI corpus:
    - *authors:* to search for a specific author or co-authors of a document
    - *title:* to search for a word appearing in the title
    - *subject:* to search for a word appearing in the human summarize of the content of the document (metadata available for the HTML documents only)
    - *keywords:* to search for a word in the human-selected keywords (metadata available for the HTML documents only)

The metadata extracted by our prototype are semi-structured *i.e.* we have coded the type of the metadata but not their components. For example, we have coded the metadata '*February 15, 2006 Wednesday*' as a date but do not break this down into the year, the month and the day. This facilitates their extraction but prevents precise request on metadata. As required by our stakeholder we will structure the date using regular expressions in the next version of the prototype.

The second TM based operators allows to search for a word (or a noun phrase) annotated by a specific TM tool where the word (or noun phrase) occurs in the content of the document. The current version of the prototype implements two operators:

- *NECanonical:* to search for a noun phrase annotated as a named entity. This operator search for the canonical form of the NE. Here is only the lowercase form of the NE.
- *NEType:* to search for a noun phrase which belong to a certain type of NE (*e.g.*

*NEType:university* would return all the documents mentioning a name of university). Because we are currently defining the types of interesting NE, this operator will be available in the next version of our prototype.

- *Term:* to search for a noun phrase annotated as a term.

These operators make possible to extend the query with semantic attributes providing a more focused or, on the opposite, a more general query.

The benefits of the recognition for the NEs' types has been illustrated with the NaCTeM's tool Kleio[13], a tool designed to process genomic documents. Names of proteins can be ambiguous with English common words like the protein '*cat'*. A classical search engine will return all the documents speaking about the protein and the animal that is more than 60,000 for a search across the whole MEDLINE[14] abstracts. With the automatic recognition of the proteins, a specific operator for searching only the document with annotated protein entities has been implemented. The query using this operator '*Protein:cat'* is obviously more accurate with 237 documents returned.

On the opposite these operators can be used to retrieve more related documents of interest. As an example, consider the automatic recognition of the names of the firms selling biometric equipments under a determined subcategory of NEs. These firms are strongly involved in the debate for the introduction of ID cards in UK. An operator defined to query this subcategory could allow a sociologist studying this debate to identify all of the documents where these firms are mentioned and relieves him/her from several queries focused on a particular firm.

We are currently discussing with our stakeholder how to qualitatively evaluate the improvement of these operators in their search activities.

## 3.5 Query results clustering and classification

### 3.5.1 A Clustering Problem

The traditional presentation of a search result returned for a free query is a list of documents with a short context where the words of the query occurred in the documents called snippets. The snippets are useful however, when a search returns numerous documents, the list of snippets is too long to be read. A solution is to cluster the documents retrieved according to the documents similarities and to associate a readable label to each cluster.

This task, referred to as the search result clustering problem, presents two issues. The number of relevant clusters is unknown. It has to be calculated in real time according to the result of the query. When the number of clusters is fixed, they are useless for the user unless readable and unambiguous tags label the clusters. To address these issues the search result clustering algorithm *Lingo* [Osinski,

[13] A web demonstrator for Kleio is available at http://www.nactem.ac.uk/software/kleio

[14] MEDLINE is an online database of citations and abstracts from the medical journals: http://www.ncbi.nlm.nih.gov/pubmed/

03] has been selected and integrated in the ASSERT search engine. The current ASSIST prototype based on the ASSERT search engine includes this component.

We describe informally the 4 stages of the pseudo code given by the author to explain this algorithm:

- Pre-processing (Stage 1): Lingo starts by automatically generating small snippets from the results of the search. An internal stopword dictionary and a stemmer are applied to improve the similarity measure in the next stages.
- Phrase extraction (stage 2): The longest and most frequent phrases appearing in the pre-processed snippets are calculated and filtered by a frequency threshold.
- Cluster label induction (stage 3): The documents are represented with words vectors during this stage. Only the most important words are present in the vectors. This importance is function of the frequency of the word and the length of the document (computed based on the *tfidf* score). The algorithm uses this representation to group the important words together according to their presence in the same documents. These groups of important words are called '*abstract concepts*'. They are ordered according to the number of words they contain. The number of clusters is a certain proportion of the top level abstract concepts. This proportion is given by the user. The abstract concepts as well as the phrases extracted in stage 2 are word vectors, so they can be compared. Using a measure of similarity, each phrase is compared with the abstract concept, and the closer phrase becomes the label of the cluster associated to the abstract concept.
- Cluster population (Stage 4): In this stage the label of all the clusters are known. Each word vector representing a document is compared using a similarity measure to the word vector representing the label of the cluster. If the similarity score is above a threshold the document is added to the cluster; otherwise, the document is added to a specific cluster labelled '*other*'.

### 3.5.2 Lingo algorithm limitations

If the problem of the number of clusters is addressed with the most simple solution, computing this number as a predefined proportion of documents, the strategy to label the cluster is innovative. To ensure that the clusters have readable labels, this algorithm starts to compute them before populating the clusters. The preliminary qualitative evaluation realized by our stakeholders shows that if most of the labels are readable and correctly describe the clusters, some of them, even if they are readable, are meaningless for them. For instance, The NCeSS corpus presents some distinctive features which lead the algorithm to meaningless titles of clusters. Several documents have the title '*Dear Sun*' or '*Letters to the editor*' because they are specifics pages of newspapers. These noun phrases are long enough to be selected as candidates and, as they appear in the title, they are favoured by the algorithm. A full evaluation of the quality of the cluster is currently in process by our stakeholders. We will detail the results of this evaluation in the next technical report.

We are considering as a future work two modifications to improve the quality of the cluster labels computed by the Lingo algorithm. The first modification is to integrate a stoplist to remove the noun phrases that are not possible terms. The second modification is to force the algorithm to consider the best terms computed by Termine as cluster labels. The current Lingo algorithm computes the terms based on their frequencies. This strategy is simpler than the hybrid strategy used by Termine which can isolate terms of better quality to title the clusters.

### 3.5.3 A classification problem

Given the problem of assigning documents to a taxonomy, two possibilities can be considered, depending on the availability of training data.

If the training data can be provided by our stakeholder, this is a classical multi-class multi-label classification problem in machine learning, The training data includes a set of documents associated with prior (already known) target information to indicate which concepts they belong to. With the given training documents, a classifier can be trained using different algorithms, such as support vector machine (SVM) [Cortes and Vapnik, 95], fisher's linear discriminant analysis (FLDA) [Fisher, 36], naïve Bayes classifiers (NBC) [Domingos and Pazzani, 97], and minimum distance classifier (MDC) [Lin and Venetsanopoulos, 93].

If it is not possible to build a training corpus, knowledge on statistics and linear algebra can be applied to conduct unsupervised learning. One possible solution is to translate the given documents and the corresponding words occurring in those documents into a subspace, by employing latent semantic analysis (LSA) [Landauer *et al.,* 98]. The predefined concepts can also be translated into the same subspace by viewing those concepts as small pseudo documents [Law *et al.* 04]. The documents and the concepts can be related to each other in this subspace with a score evaluation scheme.

It is highly probabe that the supervised learning methods using a training corpus will provide better performance than the unsupervised learning methods solely based on word (term)-document information.

## *4 Planned Activities*

Our immediate task is to extend the hierarchy of NEs using a semi-supervised machine learning based NE recognizer. When this component is fully available, we will evaluate the components of our current prototype as well as the general design of the web interfaces in collaboration with our stakeholders. This evaluation will stress the weaknesses of our prototype and characterize where our efforts should be focussed for the next version.

## *REFERENCES*

S. Kripke, *Naming and Necessity*, Harvard University Press. 1982

S. Sekine and C. Nobata, *Definition, dictionaries and tagger for Extended Named Entity Hierarchy*, Proc. Conference on Language Resources and Evaluation. 2004

S. Ananiadou and J. McNaught, (eds): *Text Mining for Biology and Biomedicine.* Artech House, London. 2006

T. Poibeau, *Extraction automatique d'information. Du texte brut au web sémantique.* Paris. Hermès. 250 pages. 2003

D. Nadeau and S. Sekine, *A Survey of Named Entity Recognition and Classification. In: Sekine, S. and Ranchhod, E. Named Entities: Recognition, classification and use*. Special issue of Lingvisticæ Investigationes. 30(1) pp. 3-26. 2007

D. Nadeau, P. Turney and S. Matwin, *Unsupervised Named Entity Recognition: Generating Gazetteers and Resolving Ambiguity.* Proc. Canadian Conference on Artificial Intelligence. 2006

D. Nadeau, *Création de surcouche de documents hypertextes et traitement du langage naturel.* Proc. Computational Linguistics in the North-East. 2005

A. Mikheev, M. Moens and C. Grover, *Named Entity Recognition without Gazetteers.* Proc. Conference of European Chapter of the Association for Computational Linguistics. 1999

Y. Sasaki, Y. Tsuruoka, J. McNaught and S. Ananiadou, *How to make the most of NE dictionaries in statistical NER*, BMC Bioinformatics, 9(Suppl 11):S5, 2008.

A. McCallum, D. Freitag and F. Pereira, *Maximum entropy Markov models for information extraction and segmentation.* Proceedings of the Seventeenth International Conference on Machine Learning, 591-598. 2000

J-D Kim, T. Ohta, Y. Tsuruoka and Y. Tateisi, *Introduction to the Bio-Entity Recognition Task at JNLPBA*. Proceeding of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-2004), 70-75. 2004

J-D Kim, T. Ohta, Y. Tateisi and J. Tsujii, *GENIA corpus – semantically annotated corpus for bio-textmining.* Bioinformatics, **19:**i180-i182. 2003

EF Tjong Kim Sang, J. Veenstra, *Representing Text Chunks.* Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics (E-99); Bergen, June 8 – 12, 173-179. 1999

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

R. Grishman. *Tipster architecture design document version 3.1.* Technical report, DARPA, 1997.

H. Han, E. Manavoglu, H. Zha, K. Tsioutsiouliklis, L. Giles, X. Zhang, *Rule-based Word Clustering for Document Metadata Extraction*, in Proceedings of the 20th Annual ACM Symposium on Applied Computing Special Track on Information Access and Retrieval (SAC-IAR'05): 1058-1062, 2005.

A. Ivanyukovich, M. Marchese, *Unsupervised Metadata Extraction in Scientific Digital Libraries Using A-Priori Domain-Specific Knowledge.* SWAP 2006

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

C. Cortes and V. Vapnik. *Support-vector networks*. Machine Learning, 20(3):273– 297, 1995.

R. A. Fisher. *The use of multiple measurements in taxonomic problems*. Annals of Eugenics, 7(2):179–188, 1936.

P. Domingos and M. J. Pazzani. *On the optimality of the simple bayesian classifier under zero-one loss*. Machine Learning, 29(2-3):103–130, 1997.

H. Lin and A. N. Venetsanopoulos. *A weighted minimum distance classifier for pattern recognition*. In Proc. of the 6th Canadian Conf. on Electrical and Computer Engineering, pages 904–907, Vancouver, BC, Canada, 1993.

T. K. Landauer, P. W. Foltz, and D. Laham, *Introduction to Latent Semantic Analysis*. Discourse Processes, 25, pp. 259-284, 1998.

S. Law, O. Jerzy, and S. Dawid, *Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition*, Advances in Soft Computing, Intelligent Information Processing and Web Mining, Proc. of the Int'l IIS: IIPWM´04 Conference, Zakopane, Poland, pp. 359-368, 2004.