Cultural Heritage Digital Resources: from Extraction to Querying

Michel Généreux
Natural Language Technology Group
University of Brighton
Umited Kingdom
M.Genereux@brighton.ac.uk

Abstract

This article presents a method to extract and query Cultural Heritage (CH) textual digital resources. The extraction and querying phases are linked by a common ontological representation (CIDOC-CRM). A transport format (RDF) allows the ontology to be queried in a suitable query language (SPARQL), on top of which an interface makes it possible to formulate queries in Natural Language (NL). The extraction phase exploits the propositional nature of the ontology. The query interface is based on the Generate and Select principle, where potentially suitable queries are *generated* to match the user input, only for the most semantically similar candidate to be *selected*. In the process we evaluate data extracted from the description of a medieval city (Wolfenbüttel), transform and develop two methods of computing similarity between sentences based on WordNet. Experiments are described that compare the pros and cons of the similarity measures and evaluate them.

1 Introduction

The CIDOC-CRM (DOERR, 2005) ontology is an ISO standard created to describe in a formal language the explicit and implicit concepts and relations underlying the documentation produced in CH. The ontology aims at accommodating a wide variety of data from the CH domain, but its sheer complexity may make it difficult for non-experts to learn it quickly, let alone use it efficiently. For others, it may even be simpler to find a way to translate automatically their data from the storage mechanism already in place into CIDOC-CRM. For practitioners unfamiliar with strict formalisms, it may be more natural to describe collections in natural language (e.g. English), and there is already an unprecedented wealth of information available on-line in natural language for almost anything, including CH. Wouldn't it be practical to be able to describe a collection of artifacts in plain English, with little or no knowledge of the CIDOC-CRM formalism, and let language technology take over and produce a CIDOC-CRM database? The principle behind that idea is based on the observation that the building blocks of the CIDOC-CRM ontology, the *triples*, have a predicative nature, which is structurally consistent with the way many natural languages are built (DOERR, 2005):

> The domain class is analogous to the grammatical subject of the phrase for which the property is analogous to the verb. Property names in the CRM are designed to be semantically meaningful and grammatically correct when read from domain to range. In addition, the inverse property name, normally given in parentheses, is also designed to be semantically meaningful and grammatically correct when read from range to domain.

A triple is defined as:

DOMAIN PROPERTY RANGE

Proceedings of the Workshop on Language Technology for Cultural Heritage Data (LaTeCH 2007), pages 41–48,
Prague, 28 June 2007.

The domain is the class (or entity) for which a property is formally defined. Subclasses of the domain class inherit that property. The range is the class that comprises all potential values of a property. Through inheritance, subclasses of the range class can also be values for that property. Example 1 is somewhat artificial, but it illustrates how triples can be extracted from natural language, where entities E48 and E53 are *Place Name* and *Place* respectively, while P1 is the property *identify*.

(1) *Rome* *identifies* *the capital of Italy.*
DOM E41 PROP P1 RANGE E1
E48 P1 E53
'Rome identifies the capital of Italy.'

The task of the natural language processing tool is to map relevant parts of texts to entities and properties in such a way that triples can be constructed (see also (SHETH, 2003; SCHUTZ, 2005; DAGAN, 2006)). In a nutshell, the Noun Clauses (NC) *Rome* and *the capital of Italy* are mapped to *Entity 48* and *Entity 53*, themselves subclasses of the domain E41 and range E1 respectively, while the Verb Clause (VC) *identifies* is mapped to *Property P1*.

On the other hand, a natural language interface (ANDROUTSOPOULOS, 1995) to query structurally complex and semantically intertwined data such as those that can be found in the archaeological domain can lighten a great deal the tasks of browsing and searching. This state of affairs is even more true for people not familiar with formal languages, as is often the case in archaeology in particular and cultural heritage in general. With the Semantic Web[1] in full development and ontologies such as CIDOC-CRM teaming together to render semantic navigation a realistic prospect, natural language interfaces can offer a welcomed simplified view of the underlying data.

One of the most important and Semantic Web oriented conceptual model available today is the CIDOC-CRM, which is becoming the new standard model to document CH data: the creation of tools ready to manage CIDOC-CRM compliant archives will be one of the most important goals of the coming years (HERMON, 2000). The full implementation of the CIDOC-CRM model is simplified today by a family of languages developed by the World Wide Web Consortium[2] and XML-based (LI, 2006). One of its most important representative is RDF[3], on top of which sits a query language such as SPARQL[4].

2 Extraction

2.1 Methodology

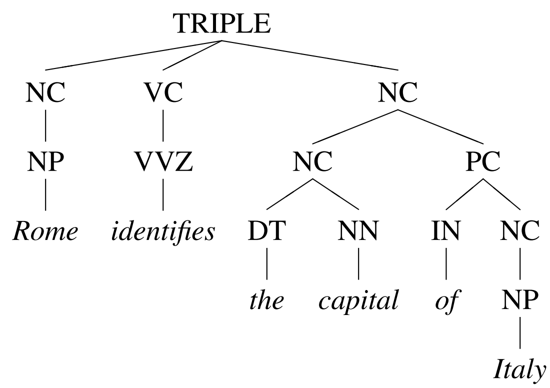

Figure 1: Linguistic parse tree for example 1.

Figure 1 suggests that all pairs of NC separated by a VC (and possibly other elements) are potentially valid CIDOC-CRM triples. Part-of-speeches (POS) and phrasal clauses can be obtained with a POS tagger and chunker[5]. To validate the triples, we must first make sure that the predicate is relevant by extracting the main verb of the verbal clause (VC) and see if its meaning is similar (synonym) to at least one of the CIDOC-CRM properties. For example, it is possible to use the verb *describe* instead of *identify*. Once a set of possible properties is identified, we must verify if the noun clauses (NC) surrounding the property are related to the DOMAIN and the RANGE of that property. To establish the relation, the first step is to identify the semantics of each NC clause. For English, a good indicator of the NC semantics is the rightmost NN in the clause, excluding any attached PC. The rightmost NN is usually the most significant: for example, in the NC *the museum artifact*, the main focus point is *artifact*, not *museum*. In figure 1 the rightmost NN of *the capital*

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

[2] W3C: http://www.w3.org/

[3] http://www.w3.org/RDF/

[4] http://www.w3.org/TR/2006/CR-rdf-sparql-query-20060406/

[5] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/

of Italy is *capital* (excluding the attached PC); this tells us that we are dealing with an object of type *capital*. The second step is to see if the type is a subclass of the DOMAIN or RANGE. Because *entity* (E1) is a hypernym of *capital*, then we conclude that the clause *the capital of Italy* is a subclass of E1:CRM Entity. What if the NC has no NN? One possibility[6] is that the clause is made up of at least one proper noun (*Rome*). To establish the type of a proper noun, we use the Web as corpus and compute a measure of *semantic association* (CHURCH, 1989) with all CIDOC-CRM classes and choose the most similar as being the type of the NC clause. This would yield the following triple:

E41		*E1*
Rome	P1	*the capital of Italy*

where E1 and E41 are the entities *Appellation* and *CRM Entity* respectively.

2.2 Extracting a triple from free text

The following experiment shows the result of extracting a triple from a textual description of the medieval city of Wolfenbüttel based on the method described previously. The document was 3922 words long with 173 sentences. The system extracted 197 intermediate triples and 79 final triples. Table 1 shows a few processing steps for the following fragment of text:

> The street's particular charm lies in its broad-faced half-timbered buildings.

In step ①, an intermediate triple is extracted from texts, then we use synonyms and hypernyms in step ② to find mappings with domains (D), properties (P) and ranges (R) of the ontology. The final triples appears in step ③. For example, *consist* is a synonym for *lie*, and *object* is an hypernym of *building*. In each case, we extracted from WordNet[7] (PEDERSEN, 2004) the synonyms and hypernyms of the three most common uses for each word (verb, noun).

[6]The other possibility, *pronouns*, is omitted for simplicity.
[7]http://wordnet.princeton.edu/

①	D	[The street's particular charm]
	P	lies in
	R	[its broad-faced half-timbered buildings]
②	D	[attribute, charm, entity, language, object]
	P	[consist]
	R	[activity, building, creation, entity, event, object]
③	D	[e13:Attribute Assignment]
	P	p9:consists of
	R	[e7:Activity]

Table 1: A triple extracted from free text.

3 Querying

3.1 NL Interface to SPARQL Querying

Our approach to the problem of mapping a query in natural language to a query expressed in a particular query language (here SPARQL) is to *generate* (BURKE, 1997) the most likely candidates and *select* the item which shows maximum semantic similarity with the input string. We now explain both steps in turn.

3.1.1 Generation

We start from two parallel grammars describing both the target query language and one or more natural languages. Here is an excerpt from one query language (SPARQL),

$$\begin{gathered} \mathit{SelectQuery} \rightarrow \mathit{Select} \left\{ \begin{array}{c} \mathit{Var+} \\ \mathit{Star} \end{array} \right\} \mathit{DC?\ WC\ SM?} \\ \mathit{DC} \rightarrow \mathit{From\ Table} \\ \mathit{WC} \rightarrow \mathit{Where?}\ \{\ \mathit{Filter}\ \} \\ \mathit{SM} \rightarrow \mathit{OrderBy\ Modifier} \\ \mathit{Star} \rightarrow \text{'}*\text{'} \\ \mathit{Select} \rightarrow \text{'}\mathit{select}\text{'} \\ \mathit{From} \rightarrow \text{'}\mathit{from}\text{'} \\ \mathit{Table} \rightarrow \text{'}< \mathit{OneTable} >\text{'} \end{gathered}$$

and part of its equivalent in natural language (here English):

$$\begin{gathered} \mathit{Select} \rightarrow \left\{ \begin{array}{c} \text{'}\mathit{select}\text{'} \\ \text{'}\mathit{show}\text{'} \end{array} \right\} \text{'}\mathit{the}\text{'}? \\ \mathit{From} \rightarrow \text{'}\mathit{from}\text{'} \end{gathered}$$

$$Star \rightarrow \left\{ \begin{array}{c} \textit{'all records'} \\ \textit{'everything'} \end{array} \right\}$$
$$OneTable \rightarrow \textit{'clients'}$$

Therefore, for a SPARQL query such as *select * from <clients> {}*, we are able to generate the equivalent in natural language: *select all records from clients*. The generation space of SPARQL and natural languages can be very large (in fact it can be infinitely large in both cases), so generation must be constrained in some way (it is in fact constrained by the size of the input string). More specifically, the grammar generates candidate strings of length to be contained between a fraction *f1* shorter and a fraction *f2* longer than the size (in meaningful words) of the input strings. Meaningful words are limited to be adjectives (tag J), nouns (tag N), verbs (tag V) and adverbs (tag RB), partly because they can be compared against each other using WordNet. The values of *f2* is usually less than the value of *f1*, but the exact values are to be determined empirically. The idea behind this is based on the general observation that queries expressed in natural languages are more likely to be redundant than underspecified. Let's look at example 2, a particular example of a user query.

(2) Could/MD you/PP show/VVP me/PP all/PDT the/DT records/NNS you/PP have/VHP please/VV ./SENT

There are three (*show*, *records* and *have*) meaningful words in 2. Assuming that we have 0.4 and 0.1 for the values of *f1* and *f2* respectively, the generator would then be constrained to produce candidate strings having a length in the range [3-0.4*3, 3+0.1*3] or [1.8, 3.3], i.e. between two and three words after rounding. The generative process must also be informed on possible values employed by the user for the sake of filtering. For example, in queries such as *Show me everything that has a salary above 500* and *Select people named Smith*, the value of the fields *salary* and *name* are respectively specified as being above *500* and *Smith*. These values are used by the generator. They are assume to be found as symbols (SYM), foreign words (FW), nouns (N), cardinal numbers (CD) or adjectives (J) in the input string. The whole generative process can be summarised as follows:

1. Compute the input query strings length *I* in meaningful word tokens and detect potential field values (SYM, FW, N, CD or J)

2. Generates candidate strings of a given language *L* with length in the range [I-f1*I, I+f2*I]

3. For each candidate string, generate the equivalent SPARQL query

The candidate strings in language *L* from step 2 are passed on to the *selection* process.

3.1.2 Selection

The *selection* process is based on a measure of similarity between the input string and candidates issued from generation. The two similarity measures we are presenting are based on an available semantic resource for English, *Wordnet*. Both measures assume that two sentences are semantically similar if they share words with similar meanings. This assumption is certainly not true in general but, in the case of database querying, we can assume that the use of metaphors, irony or contextualised expressions is relatively rare. There are different approaches to compute similarity, but we are constrained by the fact that the system must potentially analyse and compare a large number of sentences with varying lengths. The so-called *Levenshtein distance* or *edit distance* is a simple option based on dynamic programming (RISTAD, 1998). It can be transformed to become a *semantic distance*, that is, the semantic distance between two strings is given by the minimum number of operations needed to transform one string into the other, where an operation is an insertion (cost 1), deletion (cost 1), or substitution of a single word (as opposed to letters in the original edit-distance). The exact cost of substitution is given by how dissimilar a pair of words is according to WordNet (from 0 to 2). Two strings are therefore similar if they have words semantically related, with a preference for the *same word order*. This last requirement is not always acceptable for natural language, as can be illustrated by examples 3 and 4, which are clear semantic equivalent, although a measure based on the *Levenhstein distance* would be unduly penalising because of a different word order.

(3) Show me the name and salary of all clients.

(4) Look into clients and show me their name and salary.

However, the *edit distance* is computationally attractive and it is not clear whether word-order is such an important factor when querying database in natural language.

One way to have more control over word-order is to built a similarity matrix. A similarity matrix provides a pairwise measure of similarity between each word of two sentences. Let's say we want to compare the similarity of a user's sentence 5 with a candidate query 6 generated by system.

(5) Show me salaries for names Smith.

(6) Select the salary where name is Smith.

The corresponding similarity matrix is shown as table 2. Each word is assigned a part-of-speech and transformed to its base-form to simplify comparison using WordNet. The similarity values in the table are

Similarity	show	salary	name	Smith
select	25	0	0	0
salary	0	100	8	6
name	0	8	100	17
be	33	0	0	0
Smith	0	6	17	100

Table 2: Similarity matrix between two sentences

in the [0,100] range. They are computed using simple edge counting in WordNet, a technique similar to computing how two people are genetically related through their common ancestors (BUDANITSKY, 2001). Only nouns, verbs, adjectives and adverbs can be semantically related by WordNet, therefore strings are initially stripped of all other grammatical categories. For example, table 2 shows that the word *select* has a degree of similarity of 25 with show. This approach does not take on board word-order *at all*, and we introduce a slight correction for the value of each entry in the table: similarity is decreased when words appear in different positions in a string. This is a sensible compromise to consider word-order without undue penalties. This approach can be expressed as follows: similarity values are decreased by a maximum of MaxDecrease only when a pair of words are significantly distant (by factor SigDistant) in their respective position within each string. This is expressed by the following formula:

$$IF \ \frac{\mid l-c \mid}{L} > SigDistant \ THEN$$

$$Sim \leftarrow Sim * \left(1 - \frac{\mid l-c \mid}{L} * MaxDecrease\right)$$

where *l* and *c* are the line and column numbers respectively and *L* is the size of the longest string. If we set the values of SigDistant and MaxDecrease to 0.2, then table 2 is transformed to 3. In table 3,

Similarity	show	salary	name	Smith
select	25	0	0	0
salary	0	100	7	5
name	0	7	100	17
be	28	0	0	0
Smith	0	5	16	100

Table 3: Transf. sim. matrix between two sentences

we can see that the similarity between *show* and *be* has been reduced from 33 to 28. Once we have the transformed similarity matrix, we can compute the similarity between the two sentences as such. This is achieved by the following four steps:

1. Generate all possible squared (k*k) sub-matrices from the transformed similarity matrix. There are $C_n^k = \frac{n!}{k!(n-k)!}$ such matrices where k is the size of the shortest sentence and n the longest

2. Generate all possible word-pairings for each sub-matrices. This amounts to selecting elements being on a different row and column. There are k! such pairings for each $C_n^k = \frac{n!}{k!(n-k)!}$ squared sub-matrices

3. Compute the similarity of each k! word-pairs for all C_n^k sub-matrices by adding their similarity value

4. The similarity of the transformed matrix is taken to be the same as the highest among the k! word-pairs * C_n^k sub-matrices, divided (normalised) by the size of the longest string n

For our running example in table 3, step 1 yields five 4*4 sub-matrices. For each sub-matrix, there are 24 word-pairings (step 2). It is easy to see which word pairing from table 2 gives the highest similarity: (be-show,28), (salary-salary,100), (name-name,100) and (Smith-Smith,100), for a total of 328, normalised to the length of the longest string (5): 328/5 = 66. For comparison, the semantic similarity distance between the same two sentences using the edit-distance is 250, and this must be normalised to the added length of the shortest and the longest sentence, 250/(5+4) = 28. Since Levenhstein gives us a distance, we have 1-distance for similarity. The normalising factor is (longest+shortest = 5+4), since two strings completely different would necessitate k replacements and n-k insertions. The maximum cost is therefore k*2 + (n-k) = k+n.

We can get a flavour of the computational complexities involved in both measures in terms of the number of semantic similarity computations between two words (the most costly computation). The ration between these numbers for *Matrix* (n!/(n-k)!) and *Edit* (k*n) is (n-1)!/k(n-k)!. This ratio is equal or greater than 1 in all cases except when n=k=2 and n=k=3, which confirms the expected greater complexity of the *Matrix* method. For example, when two strings of 8 words (n=k=8) are compared, complexity is 64 for *Edit* and 40320 for *Matrix*.

3.2 Comparative Evaluation

In this experiment[8] we aim at evaluating and comparing the two (word-based) measures of semantic similarity between sentences previously described and based on WordNet. We will refer to these measures as *Edit* and *Matrix*. We need a reference corpus where phrases are paired as *paraphrases*, so we used the Microsoft Research Paraphrase Corpus (QUIRK, 2004), which is described by the authors as:

> . . . a text file containing 5800 pairs of sentences which have been extracted from news sources on the web, along with human annotations indicating whether each pair captures a paraphrase/semantic equivalence relationship.

[8] Values of parameters for the methods: cost of substitution = 2, word-pairings are centred, contiguous and do not exceed 7, MaxDecrease=0.2, SigDistant=0.2, method for similarity = count of edges

One of two levels of quality is assigned to each paraphrase (0 or 1). For example, phrases 7 are better paraphrases (annotated “1”) than 8 (annotated “0”).

(7) Amrozi accused his brother, whom he called ”the witness”, of deliberately distorting his evidence./ Referring to him as only ”the witness”, Amrozi accused his brother of deliberately distorting his evidence.

(8) Yucaipa owned Dominick’s before selling the chain to Safeway in 1998 for $2.5 billion./ Yucaipa bought Dominick’s in 1995 for $693 million and sold it to Safeway for $1.8 billion in 1998.

We selected random subsets of 100 pairs of good paraphrases (i.e. annotated with “1”), 100 pairs of less good paraphrases (annotated with “0”) and 100 pairs of phrases not paraphrases of each other. We computed semantic similarity for each subset using both methods. Results are presented in table 4. For each method the minimum and maximum values of similarity are reported. Variance is relatively low and both methods appear to correlate. As expected, paraphrases have higher similarity values, with type “1” values slightly ahead. Moreover, average values for paraphrases are significantly higher than for non-paraphrases, which is a sign that both methods can discriminate between semantically related sentences. When querying databases, we cannot always expect a clear front runner, but a continuum of more or less likely valuable candidates, more in line with the case of paraphrases “0”.

Compar.	Min	Avg	Max	Var	Cor
No(E)	5	12	24	0.2	0.7
No(M)	3	14	30	0.4	0.7
“0”(E)	21	57	86	1.2	0.8
“0”(M)	20	54	84	3.3	0.8
“1”(E)	35	69	94	1.9	0.6
“1”(M)	34	61	84	2.4	0.6

Table 4: Compar. eval. of the (E)dit and (M)atrix methods for types “0”, “1” and (No) paraphrases.

2-best pairs In this last experiment, 40 sets of 9 phrases are submitted to each method for evaluation. Each set includes only one pair of paraphrases: sets 1 to 20 include type “0” paraphrases, while sets 21 to 40 include type “1” paraphrases. There was no indication in the corpus that two phrases were not paraphrase of each other, so we assumed that phrases not paired as being paraphrases were not. Therefore, our random selection of non-paraphrases can be more or less dissimilar. Table 5 show the results, where underlined similarity scores are those of the actual paraphrases, and columns BEST and SECOND give the actual measures of similarity for the best match (the pair the system thinks are paraphrases) and its closest follower respectively. We can see that all 40 paraphrases were selected as the best by both methods (M and E). Numbers in bold indicate cases where methods have selected different second best. The differences between type "0" and "1" are consistent with those observed in table 4. These are very encouraging results that suggest both methods could be used in a real system.

S	Type 0				Type 1				S
E	Best		Second		Best		Second		E
T	M	E	M	E	M	E	M	E	T
1	43	54	19	20	59	48	26	17	21
2	39	38	19	14	62	94	24	17	22
3	40	59	32	21	74	90	**16**	**18**	23
4	46	65	**24**	**20**	57	86	**39**	**21**	24
5	51	57	**33**	**31**	47	47	25	19	25
6	39	43	**19**	**15**	53	54	15	11	26
7	54	70	41	39	46	60	**16**	**15**	27
8	50	59	13	9	51	79	**12**	**10**	28
9	72	78	17	20	52	62	**21**	**14**	29
10	60	67	33	23	56	60	42	29	30
11	56	78	**17**	**15**	56	52	27	26	31
12	36	50	**15**	**14**	84	79	**21**	**17**	32
13	72	80	**18**	**16**	48	60	29	27	33
14	66	68	**29**	**25**	80	79	16	13	34
15	39	65	15	12	84	87	**34**	**29**	35
16	52	58	**10**	**9**	52	77	22	14	36
17	75	71	**23**	**21**	84	82	21	18	37
18	48	53	22	19	84	87	**21**	**17**	38
19	60	60	27	19	69	71	**15**	**13**	39
20	84	63	**18**	**14**	55	80	**18**	**18**	40

Table 5: Similarity scores for each of the 2 most similar pairs of phrases as computed by M and E

3.3 Conclusions and Future Work

It is difficult to have a comprehensive evaluation of the extraction phase through standard metrics (precision, recall), since there is no benchmark for this type of analysis. A good benchmark would be a CIDOC-CRM human-annotated text. Yet we can give some evidence of the performance of the system. In our experiment, we have collected 79 final triples from a 173 sentences long document describing buildings and places of interest in a medieval city. The data was relatively clean, although punctuation was heavily used throughout the document, confusing the chunker. Despite modest results, there is no doubt that a system like this gives a head start to anyone wishing to build a collection using the CIDOC-CRM ontology. A first pass in the documentation gives a good idea of what the textual documentation is about. However, a fuller interpretation will often involve combining many triples together to form paths. Because of time restriction, we have decided to process the three most common meanings of each word that we looked up in WordNet (avoiding the need to select the correct meaning among many); this may have the side effect of lowering accuracy. Speed was not an issue without access to the Web, not an absolute necessity if we have a good thesaurus for proper nouns. Finally, we have tuned the CRM to analyse impressions of a city, which is not a domain for which the CRM is optimally intended. We conjecture that texts about museum catalogues would have yielded better results.

The approach to database querying presented in this paper demonstrates that more and more semantic resources can be used to render natural language interfaces more efficient. The semantic web provides the backbone and the technology to support complex querying of naturally complex data. Lexical resources such as WordNet makes it possible to compute semantic similarity between sentences, allowing researchers to develop original ways to semantic parsing of natural languages. Our experiments show that it is possible to map English queries to a subset of SPARQL with high level of precision and recall. The main drawback of the *Edit* method is its overemphasis on *word-order*, making it less suitable for some languages (e.g. German). The *Matrix* method is computationally greedy, and future research must investigate efficient ways of cutting down the large search space. Perhaps step 2 should limit the number of word-pairings by taking only adjacent combinations.

Another improvement might include less unconventional methods for generating the sentences such as FUF/Surge or the realiser of the LKB system, as well as the use of a corpus more relevant to CH. At this point we concede that the generation space may be problematic as input gets longer, but we conjecture that user's input should in most cases be of manageable length. Finally, more standards evaluation metrics could serve to situate the two similarity measures that are being presented with regards to more standard approaches used for the same purpose (KAUCHAK, 2006).

Finally, we have avoided the issue raised by polysemic words by considering only the most common senses found in WordNet, so the approach would be well complemented by contribution from the field of Word-Sense Disambiguation (WSD).

Acknowledgement

This work has been conducted as part of the EPOCH network of excellence (IST-2002-507382) within the IST (Information Society Technologies) section of the Sixth Framework Programme of the European Commission. Thank you to the reviewers for useful comments.

References

ANDROUTSOPOULOS I., RITCHIE G., THANISCH P. (1995). *Natural language interfaces to databases - an introduction*. Journal of Language Engineering, 1(1), 29.

BUDANITSKY A., HIRST G. (2001). *Semantic distance in wordnet : an experimental, application-oriented evaluation of five measures*. In NAACL 2001 Workshop on WordNet and Other Lexical Resources, Pittsburgh.

BURKE R.D., HAMMOND K.J., KULYUKIN V., LYTINEN S.L., TOMURO N., SCHOENBERG. S. *Question answering from frequently asked question files - experiences with the faq finder system*. AI Magazine, 18(2), 57.

CHURCH K.W., HANKS P. (1989) *Word association norms, mutual information, and lexicography*. In Proc. of the 27th. Annual Meeting of the ACL Vanvouver, B.V., 1989), pp. 76-83.

CRESCIOLI M., D'ANDREA A., NICCOLUCCI F. (2002). *XML Encoding of Archaeological Unstructured Data. In G. Burenhault (ed.), Archaeological Informatics : Pushing the envelope*. In Proc. of the 29th CAA Conference, Gotland April 2001, BAR International Series 1016, Oxford 2002, 267-275.

DAGAN I., GLICKMAN O., MAGNINI B. (2006). *The PASCAL Recognising Textual Entailment Challenge*. Lecture Notes in Computer Science, Volume 3944, Jan 2006, Pages 177 - 190.

DOERR M. (2005) *The CIDOC CRM, an Ontological Approach to Schema Heterogeneity*. Semantic Interoperability and Integration. Dagstuhl Seminar Proceedings, pp. 1862-4405. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany.

HERMON S., NICCOLUCCI F. (2000). *The Impact of Web-shared Knowledge on Archaeological Scientific Research*. Proc. of Intl CRIS 2000 Conf., Helsinki, Finland, 2000.

KAUCHAK D., BARZILAY R. (2006). *Paraphrasing for Automatic Evaluation*. In Proc. of NAACL/HLT, 2006.

LI Y., YANG H., JAGADISH H. (2006). *Constructing a generic natural language interface for an xml database*. International Conference on Extending Database Technology

PEDERSEN T., PATWARDHAN S.,MICHELIZZI J.(2004). *Wordnet::Similarity - Measuring the Relatedness of Concepts*. In Nineteenth National Conference on Artificial Intelligence (AAAI-04), San Jose, CA. (Intelligent Systems Demonstration).

QUIRK C., BROCKETT C., DOLAN W.B. (2004). *Monolingual Machine Translation for Paraphrase Generation*. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona Spain.

RISTAD E.S., YIANILOS P. N. (1998). *Learning string-edit distance*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(5), 522.

SCHUTZ A., BUITELAR P. (2005). *RelExt: A Tool for Relation Extraction in Ontology Extension*. In: Proc. of the 4th International Semantic Web Conference, Galway, Ireland, Nov. 2005.

SHETH A. (2003) *Capturing and applying existing knowledge to semantic applications*. Invited Talk "Sharing the Knowledge" - International CIDOC CRM Symposium. March 2003. Washington DC.

All web references visited on 02-05-2007.