Question Answering using Semantic Annotation

Lili Aunimo and Reeta Kuuskoski
Department of Computer Science
University of Helsinki, P.O. Box 68
FIN-00014 UNIVERSITY OF HELSINKI, Finland
`aunimo|rkuuskos@cs.helsinki.fi`

Abstract

This paper presents a question answering (QA) system called *Tikka*. *Tikka's* approach to QA relies heavily on the semantic annotation of text documents and on the usage of answer extraction patterns. In this way, *Tikka* applies to QA pattern-based techniques traditionally used in named entity recognition and information extraction. In the experiments presented in this paper, *Tikka's* performance is evaluated in the following tasks: monolingual Finnish and French and bilingual Finnish-English QA. Its performance in the monolingual tasks is near the average when it is compared with the QA systems' performance that participated in the monolingual French task. In the monolingual Finnish task, *Tikka* was the only participating system.

Categories and Subject Descriptors

]H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval; H.3.4 Systems and Software;]H.5 [**Information Interfaces and Presentation**]: H.5.2 User Interfaces: Natural Language

Keywords

Question Answering, Evaluation, Experimentation, Multilingual Information Access

1 Introduction

Question answering (QA) is a task that aims beyond document retrieval and towards natural language understanding. The task and its evaluation in the CLEF 2005 Multilingual Question Answering Track is described in the overview paper [6]. Our approach to QA relies heavily on the semantic annotation of text documents and on the usage of answer extraction pattern prototypes. In this way, we apply pattern-based techniques traditionally used in named entity recognition and information extraction to QA.

Figure 1 shows the system architecture of our QA system *Tikka*. Out of the five modules of the *question analysis* component, only the *question classifier* and the *topic and target extractor* are used to process both Finnish and French. The *syntactic parser* is used only for Finnish, the *semantic annotator* only for French, and the *translator* only for performing translation from Finnish to English. The *answer extraction* component can handle English, Finnish and French. The input to the system is a question in Finnish or French. The *question analysis* component forms the query terms for document retrieval, determines the class of the question and its topic and target words, and passes these on to the *answer extraction* component. The *answer extraction* component returns an answer to the question. Both of these components are described in detail in the following sections.

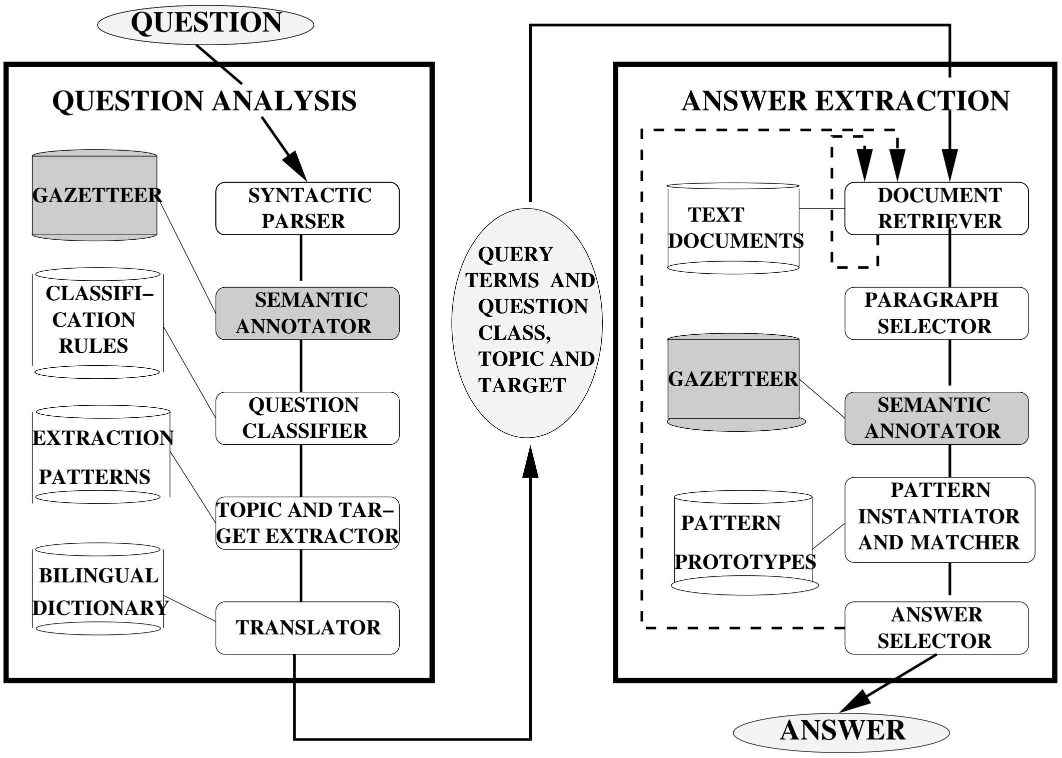

Figure 1: The system architecture of Tikka. Tikka has two main components: question analysis and answer extraction. Both components use the same semantic annotator, which is illustrated by gray in the figure. The left hand side of each component lists the databases used by it. The rectangles on the right hand side illustrate the software modules.

2 Question Analysis

The question analysis component of the QA system consists of five software modules: 1) the syntactic parser for Finnish, 2) the semantic annotator, which is detailed in Section 4, 3) the question classifier, 4) the topic and target extractor and 5) the translator, which is described in the system description of the previous version of *Tikka*, that participated in QA@CLEF 2004 [1]. All these modules, along with the databases that they use, are illustrated in Figure 1.

Table 1 shows through an example how question analysis is performed. First, a natural language question is given as input to the system, for example: *D FI EN Mikä on WWF?* [1]. Next, the Finnish question is parsed syntactically and the French question is annotated semantically. Then both questions are classified according to the expected answer type, and the topic and target words are extracted from them. The expected answer types are determined by the multinine corpus, and they are: *LOCATION, MEASURE, ORGANIZATION, OTHER, PERSON* and *TIME*. The target words are extracted or inferred from the question and they further restrict the answer type, e.g. age, kilometers and capital city[2].The topic words are words extracted from the question that in a sentence containing the answer to the question carry old information. For example, in the question *What is WWF?*, *WWF* is the topic because in the answer sentence *WWF is the World Wide Fund for Nature.*, *WWF* is the old information and *the World Wide Fund for Nature* is the new information. The old and new information of a sentence are contextually established [9]. In our case, the question is the context. In *Tikka*, topic words are useful query terms along with the target words, and they are also used to fill slots in the answer pattern prototypes.

[1]D stands for a definition question and FI EN means that the source language is Finnish and the target language is English. In English, the question means *What is WWF?*

Module	**Example**		
	English	**Finnish**	**French**
	D FI EN Mikä on WWF ?	D FI FI Mikä on WWF?	D FR FR Qu'est-ce que la WWF?
(1) Parser	1 Mikä mikä subj:>2 &NH PRON SG NOM 2 on olla main:>0 &+MV V ACT IND PRES SG3 3 WWF wwf &NH N		N/A
(2) Semantic Annotator	N/A		Qu'est-ce que <organization> la WWF</organization>?
(3) Classifier	Organization		
(4) T & T Extractor	Topic: WWF Target: N/A		
(5) FI → EN	WWF	N/A	

Table 1: The availability and output of the five modules of question analysis illustrated with the same example sentence for the target languages English, Finnish and French. Answer extraction with these same questions is illustrated in Table 2.

3 Answer Extraction

The answer extraction component consists of five software modules: 1) the document retriever, 2) the paragraph selector, 3) the semantic annotator, 4) the pattern instantiator and matcher and 5) the answer selector. All these modules, along with the databases that they use, are illustrated in Figure 1. The dotted arrows that go from the document retriever back to itself as well as from the answer selector back to the document retriever illustrate that if no documents or answers are found, answer extraction starts all over.

3.1 An Example

Module	**Example**		
	English	**Finnish**	**French**
	Query terms: WWF, Topic: WWF, Target: N/A		
(1) Document retriever	22 docs retrieved 22 docs inspected	76 docs retrieved, 30 docs inspected	313 docs retrieved, 10 docs inspected
(2) Paragraph selector	70 paragraphs selected	99 paragraphs selected	39 paragraphs selected
(3) Semantic Annotator	See Table 4		
(4) Pattern I & M	0 patterns 0 matches	12 instantiated patterns match 7 different answers	18 instantiated patterns match 4 different answers
(5) Answer selector	nothing to choose from	chooses the answer with the highest score, 18	chooses the answer with the highest score, 8
	0 NIL	0.25 AAMU19950818-000016 Maailman Luonnon Säätiö	0.75 ATS.940527.0086 le Fonds mondial pour la nature

Table 2: The output of the five different modules of answer extraction illustrated with examples. The examples are the same as in Table 1, and the processing in this table is a continuation of the question analysis illustrated in that table.

Table 2 shows through an example how answer extraction is performed. First, the question analysis passes as input to the component the query words, and the topic and target of the question. Next, document retrieval is performed using the query terms. The parameter settings of the document retrieval engine are determined by the number of times document retrieval has been performed for the question at hand. If the document retrieval succeeds, the paragraphs containing at least one query word are filtered out for further processing and they are annotated semantically. After that, a class-specific set of pattern prototypes is instantiated with the topic word and with a possibly existing target word. Each pattern prototype has a score, which reflects its accuracy. The score ranges between 1 and 9. Instantiated patterns are then matched against semantically annotated paragraphs, and answer candidates are extracted. For the example illustrated in Table 2, the pattern prototype and the corresponding instantiated pattern that matches the Finnish answer is:

```
((<[a-z]+>[^<>]+<\/[a-z]+> )+)\( (<[a-z]+>)?TOPIC(<\/[a-z]+>)? \)Score:9
((<[a-z]+>[^<>]+<\/[a-z]+> )+)\( (<[a-z]+>)?Wwf(<\/[a-z]+>)? \)Score:9
```

The text snippet that matched the above pattern is in Table 4. (The patterns are case insensitive.) Only at least partly semantically annotated candidates can be extracted. The score of a unique answer candidate is the sum of the scores of the patterns that extracted the similar answer instances, or more formally:

$$score(answer) = \sum_{i \ in \ A} patternScore(i), \tag{1}$$

where A is the set of similar answers and $patternScore(i)$ is the score of the pattern that has matched i in text. The confidence value of a non-NIL answer candidate is determined by the candidate's score and by the total number of candidates. This is illustrated in Figure 2. For example, if the total number of candidates is between 1 and 5, and the score of the candidate is 17 or greater, confidence is 1, but if the score of the candidate is between 1 and 16, confidence is 0.75. If the confidence score 1 is reached, the answer is selected and no further answers are searched. Otherwise, all paragraphs are searched for answers, and the one with the highest score is selected. Since the confidence of the answer for Finnish in Table 2 is 0.25, and the score of the answer is 18, we can deduce that the number of answer candidates is at least 11.

If document retrieval does not return any documents, or no answer is extracted from the paragraphs, *Tikka* has several alternative ways in which to proceed, depending on which task it is performing and how many times document retrieval has been tried. This is illustrated in Table 5. As can be seen from the figure, if no documents are retrieved in the first iteration, the parameter settings of the retrieval engine are altered and document retrieval is performed again in the monolingual Finnish and bilingual Finnish-English tasks. However, in the monolingual French task the system halts and returns NIL with a confidence of 1 as an answer. In the monolingual Finnish task, the system halts after the second try, but in the bilingual English-Finnish task, document retrieval is performed for a third time if either no documents are retrieved or no answer is found. Alternatively, in the monolingual Finnish and English tasks, if documents are retrieved, but no answers are found after the first try, document retrieval is tried once more. In all tasks, the system returns a confidence value of 1 for the NIL answer if no documents are found and a confidence value of 0 for the NIL answer if documents are found but no answer can be extracted.

3.2 Document Retrieval

The document retrieval module of *Tikka* consists of the vector space model [10] based search engine Lucene [2] and of the document indices for English, Finnish and French newspaper text built using it. *Tikka* has one index for the English document collection, two indices for the Finnish document collection and two for the French document collection. In each of the indices, one newspaper article forms one document. The English index (enstem) is a stemmed one. It is stemmed using

[2]http://lucene.apache.org/java/docs/index.html

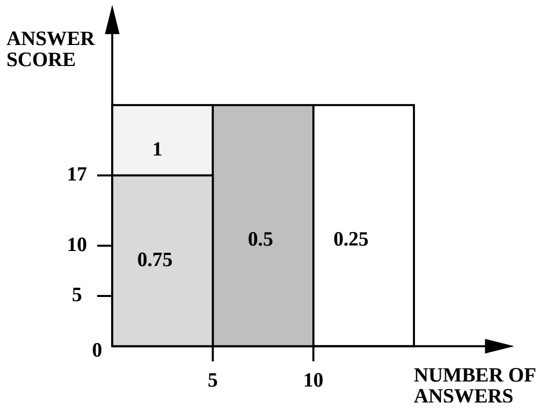

Figure 2: The confidence value of a non-NIL answer is a function of the answer's score and the number of unique answer candidates.

the implementation of Porter's stemming algorithm [7] included in Lucene. One index (filemma) to the Finnish collection is created using the lemmatized word forms as index terms. The Connexor's parser is used for the lemmatization. The other Finnish index (fistem) consists of stemmed word forms. The stemming is done by Snowball [8] project's [3] stemming algorithm for Finnish. A Snowlball stemmer is also used to create one of the indices for French (frstem). The other French index (frbase) is built using the words of the documents as such. This index is case-insensitive.

In the document retrieval phase, Lucene determines the similarity between the query (q) and the document (d) using the formula presented in Equation 2 [4].

$$similarity(q,d) = \sum_{t\ in\ q} tf\ (t\ in\ d) \cdot idf(t) \cdot boost(t.field\ in\ d) \cdot lengthNorm(t.field\ in\ d), \quad (2)$$

where tf is the term frequency factor for the term t in the document d, and $idf(t)$ is the inverse document frequency of the term. The factor *boost* adds more weight to the terms appearing in a given field, and it can be set at indexing time. The last factor is a coefficient that normalizes the score according to the length of the field. After all the scores regarding a single query have been calculated, they are normalized from the highest score if that score is greater than 1. Since we do not use the field specific term weighting, the two last terms of the formula can be discarded, and the formula is equal to calculating the dot product between a query with binary term weights and a document with $tfidf$ [5] term weights.

Lucene does not use the pure boolean information retrieval (IR) model, but we model the conjunctive boolean query by requiring all of the query terms to appear in each of the documents in the result set. This differs from the pure boolean IR model in that the relevance score for each document is calculated according to Equation 2, and the documents are ordered according to it. The ordering is important in *Tikka*. This is what the term boolean means in Figure 5. In the same figure, the term ranked means a normal Lucene query where all of the query words are not required to appear in the retrieved documents.

4 Semantic Annotation

Semantic annotation is in many ways a similar task to named entity recognition (NER). NER is commonly done based on preset names lists and patterns [11] or using machine learning techniques [3]. Our method relies on the first method. The main difference between NER and semantic

[3]http://snowball.tartarus.org/

annotation is that the first one aims at recognizing proper names whereas the second aims at recognizing both proper names and common nouns.

In *Tikka*, French questions and selected paragraphs from the search results are annotated semantically. We have 14 semantic classes that are listed in table 3. To the classes consisting mainly of proper nouns, some common nouns are added in order to be able to analyze the questions correctly. For instance, in the gazetteers of the class *organization*, there are proper names denoting companies (*IBM, Toyota*), but also some common nouns referring to organizations in each language (*school, bank, union*). As can be seen from Table 3, the *organization* gazetteer in English is significantly shorter than those in other two languages. This is due to the NER from Connexor that is used in addition to our own semantic annotator.

Class	**English**	**French**	**Finnish**	**Class**	**English**	**French**	**Finnish**
person	3704	3704	3704	unit	31	35	44
country	265	215	252	measure	51	50	34
language	109	79	637	award	15	15	7
nationality	57	177	85	color	22	20	29
capital	277	211	277	profession	95	246	127
location	5339	5440	5314	time	56	38	38
organization	37	968	212	event	29	21	15

Table 3: The semantic classes and the number of items in the gazetteers for each language.

The semantic annotator uses a window of two words for identifying the items to be annotated. In that way we can only find the entities consisting of one or two words. The external NER that is used in the English annotation is able to identify person names, organizations and locations. Hence, there are no limitations on the length of entities on these three classes in English. For Finnish, we exploit Connexor's syntactic parser for part of speech recognition to eliminate the words that are not nouns, adjectives or numerals. For French, the semantic annotator builds solely on the text as it is without any linguistic analysis.

In the text to be annotated, persons are identified based on a list of first names and the subsequent capital word. The subsequent capital words are added to the list of known names in the document. In this way the family names appearing alone later in the document can also be identified to be names of a person. The class *location* consists of names of large cities that are not capitals and of the names of states and other larger geographical items. To the class *measure* belong numerals and numeric expressions, for instance *dozen. Unit* consists of terms such as *percent, kilometer.* The *event* class is quite heterogeneous, since to it belong terms like *Olympics, Christmas, war* and *hurricane. Time* gazetteer lists time related terms, the names of the months, week days etc. Example annotations for each of the languages can be seen in Table 4.

5 Analysis of Results

Tikka was evaluated by participating in the monolingual Finnish and French tasks and in the bilingual Finnish-English task. The evaluation results are described in detail in Section 4 (Results) of the QA track overview paper [6]. In each of the tasks, two different parameter settings (run 1 and run 2) for the document retrieval component were tested. These settings are listed in Table 5. The results of the runs are shown in Figure 3. We can observe that the difference between runs is not very big for the French monolingual run. The accuracy of the artificial combination run [4] (C) is not much higher than that of the the French monolingual run 1, which means that almost

[4] In this case, the artificial combination run represents a run where the system is somehow able to choose for each question the better answer from the two answers provided by the runs 1 and 2. For more information on the combination runs, see the track overview paper [6].

Lang.	Example
English	The <organization>World Wide Fund for Nature</organization> (<organization>WWF</organization>) reported that only <measure>35,000</measure> to <measure>50,000</measure> of the species remained in mainly isolated pockets .
Finnish	<organization>Maailman Luonnon Säätiö</organization> (<ne>WWF</ne>) vetoaa kaikkiin <country>Suomen</country> metsstjiin , ett ei <person>Toivoa</person> ja sen perhett ammuttaisi niiden <unit>matkalla</unit> toistaiseksi tuntemattomille talvehtimisalueille .
French	<ne>La</ne> dcision de <organization>la Commission</organization> baleinière internationale (<ne>CBI</ne>) de créer un sanctuaire pour les cétacés est ”une victoire historique” , a commenté <time>vendredi</time> <ne>le Fonds</ne> mondial pour la nature (<organization>WWF</organization>)

Table 4: Examples of semantically annotated text snippets in English, Finnish and French, retrieved from newspaper text and answering the question *What is WWF?*. It is question number 136 and 278 in the multinine corpus [6] and it is used as an example in the Tables 1 and 2.

all answers given by the runs are equal. On the contrary, there is a difference of 4 points between the accuracies of the two monolingual Finnish runs, and in addition, as the difference between run 1 and the combination run for Finnish is 3.5 points, we can conclude that some of the correct answers returned by run 2 are not included in the set of correct answers given by run 1. This means that the different parameter settings of the runs produced an effect on *Tikka's* overall performance. Between the runs, both the parameters for type of index and the maximum number of documents were altered. In the bilingual Finnish-English task, some difference between the runs can be observed, but the difference is not as big as in he monolingual Finnish task.

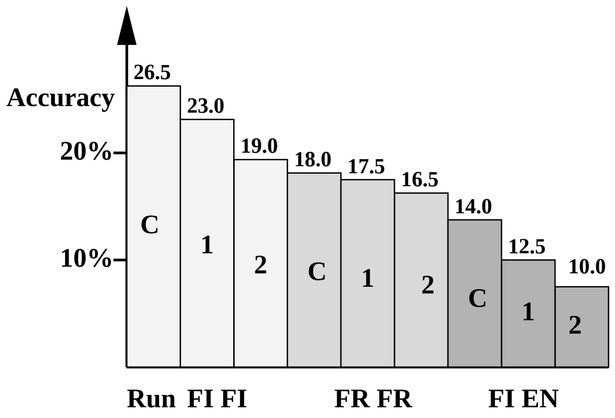

Figure 3: A histogram showing the percentage of correct answers (i.e. the accuracy) in *Tikka's* submitted test runs and in the artificial combination run. The very light gray represents the monolingual runs for Finnish, the a little bit darker gray represents the monolingual French runs and the darkest gray represents the bilingual Finnish-English runs.

6 Conclusions and Future Work

Tikka is a QA system that uses pattern-based techniques to extract answers from text. In the experiments presented in this paper, its performance is evaluated in the following tasks: monolingual Finnish and French and bilingual Finnish-English QA. Its performance in the monolingual tasks is near the average when it is compared with the QA systems' that participated in the monolingual French task. In the monolingual Finnish task, *Tikka* was the only participating system.

Table 5: The parameters for document retrieval used in different runs and in different iterations of the same run. *MinS* stands for the minimum similarity value between query and document and *MaxD* stands for the maximum number of documents to be retrieved. The maximum number of iterations is in monolingual Finnish runs 2, in monolingual French runs 1, and in Bilingual English runs 3.

Monolingual Finnish				
Iterations	**Run id: FI FI 1**			
	index	**query**	**minS**	**maxD**
(1) $\rightarrow$	filemma	boolean	0,65	30
(2) *if no documents* $\rightarrow$	fistem	ranked	0,65	20
(2) *else if no answers* $\rightarrow$	fistem	ranked	0,3	20
Iterations	**Run id: FI FI 2**			
(1) $\rightarrow$	fistem	boolean	0,65	30
(2) *if no documents* $\rightarrow$	filemma	ranked	0,65	20
(2) *else if no answers* $\rightarrow$	filemma	ranked	0,3	10
Monolingual French				
Iterations	**Run id: FR FR 1**			
	index	**query**	**minS**	**maxD**
(1) $\rightarrow$	frstem	boolean	0,65	NONE
Iterations	**Run id: FR FR 2**			
(1) $\rightarrow$	frbase	boolean	0,26	10
Bilingual Finnish-English				
Iterations	**Run id: FI EN 1**			
	index	**query**	**minS**	**maxD**
(1) $\rightarrow$	enstem	boolean	0,65	100
(2) *if no documents* $\rightarrow$	enstem	ranked	0,5	20
(3) *if no answers* $\rightarrow$	enstem	ranked	0,3	20
Iterations	Run id: FI EN 2			
(1) $\rightarrow$	enstem	boolean	0,55	100
(2) *if no documents* $\rightarrow$	enstem	ranked	0,5	20
(3) *if no answers* $\rightarrow$	enstem	ranked	0,2	20

In the future, as the document databases are not very big (about 1.6 GB), the documents could be annotated semantically before indexing. This would speed up the interactive processing time and the semantic classes could be used as fields in index creation. In addition, indexing based on paragraph level instead of document level might raise the ranking of the essential results and it would speed up the processing time of the interactive phase.

7 Acknowledgements

The authors thank Connexor Ltd [5] for providing the Finnish and English parsers and NE reconizers, Kielikone Ltd [6] for providing the bilingual dictionary and Juha Makkonen for providing the question classifier for Finnish.

[5]http://www.connexor.com

[6]http://www.kielikone.fi/en

References

[1] Lili Aunimo, Reeta Kuuskoski, and Juha Makkonen. Finnish as Source Language in Bilingual Question Answering. In C. Peters, P. D. Clough, G. J. F. Jones, J. Gonzalo, M. Kluck, and B. Magnini, editors, *Multilingual Information Access for Text, Speech and Images: 5th Workshop of the Cross-Language Evaluation Forum, CLEF 2004, Bath, UK, September 15-17, 2004, Revised Selected Papers*, volume 3491 of *Lecture Notes in Computer Science*. Springer Verlag, 2005.

[2] Lili Aunimo, Juha Makkonen, and Reeta Kuuskoski. Cross-language Question Answering for Finnish. In Eero Hyvönen, Tomi Kauppinen, Mirva Salminen, Kim Viljanen, and Pekka Ala-Siuru, editors, *Proceedings of the* 11^{th} *Finnish Artificial Intelligence Conference STeP 2004, September 1-3, Vantaa, Finland*, volume 2 of *Conference Series – No 20*, pages 35–49. Finnish Artificial Intelligence Society, 2004.

[3] Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. An algorithm that learns what's in a name. *Machine Learning*, 34(1-3):211 – 231, 1999.

[4] Erik Hatcher and Otis Gospodnetić. *Lucene in Action.* Manning Publications Co., 2004.

[5] Karen Sparck Jones. A statistical interpretation os term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21, 1972.

[6] B. Magnini, A. Vallin, , L. Aunimo, C.Ayache, G. Erbach, A. Penas, M. de Rijke, P. Rocha, K. Simov, and R. Sutcliffe. Overview of the CLEF 2005 Multilingual Question Answering Track. In Carol Peters and Francesca Borri, editors, *Proceedins of the CLEF 2005 Workshop*, Vienna, Austria, sept 2005.

[7] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[8] M.F. Porter. Snowball: A language for stemming algorithms, 2001. Available at http://snowball.tartarus.org/texts/introduction.html[22.8.2005].

[9] R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. *A Comprehensive Grammar of the English Language.* Longman, 1985.

[10] Gerard Salton. *The SMART Retrieval System: Experiments in Automatic Document Processing.* Prentice Hall, 1971.

[11] Martin Volk and Simon Clematide. Learn - Filter - Apply - Forget. Mixed Approaches to Named Entity Recognition. In *Proceedings of the 6th International Workshop of Natural Language for Information Systems*, Madrid, Spain, 2001.