University of Sheffield TREC-9 Q & A System

Sam Scott* and Robert Gaizauskas
{s.scott,r.gaizauskas}@dcs.shef.ac.uk

Department of Computer Science
University of Sheffield
Regent Court, Portobello Road
Sheffield S1 4DP UK

1 Introduction

The system entered by the University of Sheffield in the question answering track of TREC-9 represents a significant development over the Sheffield system entered for TREC-8 [6] and, satisfyingly, achieved significantly better results on a significantly harder test set. Nevertheless, the underlying architecture and many of the lower level components remained the same. The essence of the approach is to pass the question to an information retrieval (IR) system which uses it as a query to do passage retrieval against the test collection. The top ranked passages output from the IR system are then passed to a modified information extraction (IE) system. Syntactic and semantic analysis of these passages, along with the question, is carried out to identify the "sought entity" from the question and to score potential matches for this sought entity in each of the retrieved passages. The five highest scoring matches become the system's response.

2 System Description

2.1 Overview

The key features of the system setup, as it processes a single question, are shown in Figure 1. Firstly, the (indexed) TREC document collection and the question are passed to an IR system which treats the question as a query and returns top ranked passages from the collection. As the IR system we used the Okapi system [8][1]. Following this, the top ranked passages are run through a text filter to remove certain text formatting features which cause problems for downstream components. Finally, the question itself and the filtered top ranked passages are processed by a modified version of the LaSIE information extraction system [5], which we refer to below as QA-LaSIE. This yields a set of top ranked answers which are the system's overall output.

The reasoning behind this choice of architecture is straightforward. The IE system can perform detailed linguistic analysis, but is quite slow and could not process the entire TREC collection for each query, or even realistically pre-process it in advance to allow for reasonable question

*Current address: Institute for Interdisciplinary Studies, Carleton University, Ottawa, K1S 5B6, Canada. sscott@ccs.carleton.ca.

[1]Software available at: http://dotty.is.city.ac.uk/okapi-pack/. For TREC-8 we both used the NIST-supplied top documents and passages from UMass's INQUERY system [2] which UMass kindly provided for us. Our switch from INQUERY to Okapi was prompted by the acquisition of Okapi and Okapi expertise in-house.

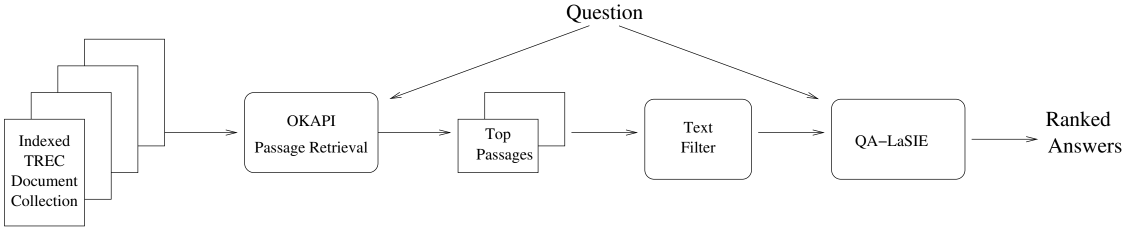

Figure 1: System Setup for the Q & A Task

answering performance during the test run. IR systems on the other hand are designed to process huge amounts of data. By using an IR system as a filter to an IE system we hope to benefit from the respective strengths of each.

In the next section we describe how we have parameterised Okapi for the QA task. The following section briefly describes the base LaSIE system and the succeeding section describes how it has been modified and extended to yield QA-LaSIE. Of the text filtering module we say no more, as it is of little intrinsic interest and was simply a convenient way of avoiding modifications to various components in LaSIE to deal with idiosyncrasies in the TREC collection texts.

2.2 Okapi

The Okapi IR system is based on the probabilistic retrieval model [7]. Since we used the system "off the shelf", we discuss here only the parameterisation adopted for the QA task, and not the underlying model. Aside from using a slightly modified version of the stop list provided with Okapi, no parameterisation of the indexing process took place. However, to utilise the passage retrieval capabilities of Okapi a number of parameters need to be set for the searching process. We set the minimum number of paragraphs to be returned to 1, the maximum number to 3, and the paragraph step unit to 1 (this parameter determines how much the sliding passage window moves between comparisons in the passage ranking process). Determining that the maximum number of paragraphs per passage should be 3 was a matter of some experimentation, and it interacted with the decision about how many passages to select per question (because of processing times in the IE system).

Our experimentation was carried out using the TREC-8 QA track question set, but the TREC-9 document collection (the latter is a superset of the TREC-8 collection). Looking at the top 5, 10 and 20 documents returned for each question we discovered that these document sets contained answers for 160, 175 and 184 of the 198 TREC-8 questions respectively. We then tried experimenting with passage retrieval using the Okapi default settings for minimum passage length (1 paragraph) and maximum passage length (20 paragraphs), and keeping the best passage only if the retrieval engine's score for it was higher than for the entire document. For 158 of the questions this resulted in an answer being found in the top 5 passages – a loss of only 2 over the full document approach. Deciding, therefore, that passage retrieval was worthwhile, we experimented with the maximum passage length parameter. By reducing it to 3 paragraphs, and always preferring best passages to full documents, even if the full document score was higher, we discovered no answers were lost (i.e. the top 5 best passages per question still contained answers for 158 of the questions). Furthermore, running the passages of maximum length 3 through the QA-LaSIE system led to considerably higher mean reciprocal rank (MRR scores) than using the full documents, presumably because there were fewer distractors.

Given the significantly smaller amount of text to be processed by QA-LaSIE using passages of at most 3 paragraphs, we were encouraged to examine more top passages per question. By considering the top 20 best passages we discovered that 164 questions had answers in the retrieved passage sets; and the MRR scores of QA-LaSIE against these were higher than for the top 5 best passages per question. These then were the parameters we finally settled upon for the TREC-9 evaluation: top 20 passages per question with maximum passage length 3 paragraphs.

2.3 LaSIE

The LaSIE system used to perform detailed question and text analysis is largely unchanged in architecture from the IE system as entered in the last Message Understanding Conference evaluation (MUC-7) evaluation [5]. The principal components of the LaSIE system are the first eight modules shown in Figure 2 as executed interactively through the GATE Graphical Interface [3]. The system is essentially a pipeline of modules each of which processes the entire text before the next is invoked. The following is a brief description of each of the component modules in the system:

Tokenizer Identifies token boundaries (as byte offsets into the text) and text section boundaries (text header, text body and any sections to be excluded from processing).

Gazetteer Lookup Identifies single and multi-word matches against multiple domain specific full name (locations, organisations, etc.) and keyword (company designators, person first names, etc.) lists, and tags matching phrases with appropriate name categories.

Sentence Splitter Identifies sentence boundaries in the text body.

Brill Tagger [1] Assigns one of the 48 Penn TreeBank part-of-speech tags to each token in the text.

Tagged Morph Simple morphological analysis to identify the root form and inflectional suffix for tokens which have been tagged as noun or verb.

Parser Performs two pass bottom-up chart parsing, pass one with a special named entity grammar, and pass two with a general phrasal grammar. A 'best parse' is then selected, which may be only a partial parse, and a predicate-argument representation, or quasi-logical form (QLF), of each sentence is constructed compositionally.

Name Matcher Matches variants of named entities across the text.

Discourse Interpreter Adds the QLF representation to a semantic net, which encodes the system's background world and domain knowledge as a hierarchy of concepts. Additional information inferred from the input using this background knowledge is also added to the model, and coreference resolution is attempted between instances mentioned in the text, producing an updated discourse model.

For standard IE template filling tasks, a final Template Writer module reads the discourse model produced by the Discourse Interpreter, derives template slots fills and writes out the filled templates.

2.4 QA-LaSIE

The QA-LaSIE system takes a question and a set of passages delivered by the IR system and returns a ranked list of proposed answers for the question. Figure 2 shows the end-to-end QA-LaSIE system as entered in TREC-9. Four key adaptations have been made to move from the base IE system described in the previous section in a system capable of carrying out the QA task:

1. a specialised grammar was developed to analyse questions and added to the parser;

2. the discourse interpreter was modified to allow the QLF representation of each question to be matched against the discourse model of a candidate answer text, using the coreference mechanism;

3. an answer identification procedure which scored all discourse entities in each candidate text as potential answers was added to the discourse interpreter;
4. a TREC Question Answer module was added to examine the discourse entity scores across all passages, determine the top 5, and then output the appropriate answer text.

Further detailing these alterations will not communicate the essence of the QA-LaSIE approach to question answering. Therefore, in the following subsections we describe the key processes involved in the QA-LaSIE approach to question answering, including, where necessary, further information about the general approach to text processing taken in LaSIE.

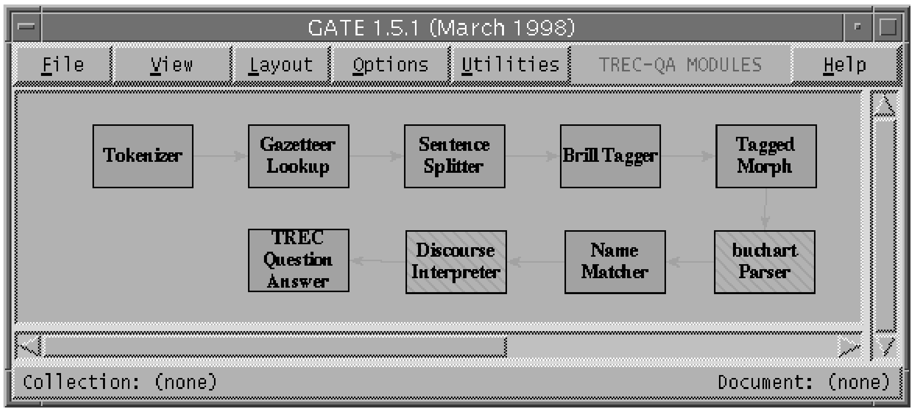

Figure 2: QA-LaSIE System Modules

2.4.1 Parsing: Syntactic and Semantic Interpretation

In the LaSIE approach, both candidate answer passages and questions are parsed using a unification-based feature structure grammar. As input the parser receives one sentence at a time and along with the original words of that sentence also receives: a part-of-speech tag from the Penn tagset for each word, morphological information for each noun and verb (word root plus affix), and zero or more phrases tagged as named entities. As output the parser produces as representation of the sentence in a "quasi-logical form" – a predicate-argument representation that stands somewhere between the surface form of the sentence and a fully interpreted semantic representation in a standard logical language. In particular the QLF representation defers issues of quantifier scoping and of word sense disambiguation.

To take a simple example, the sentence fragment *Morris testified that he released the worm ...* is parsed and transduced to the representation

```
person(e1), name(e1,'Morris'), gender(e1,masc), uncertain(e1,person),
testify(e2), time(e2,past), aspect(e2,simple), voice(e2,active),
lsubj(e2,e1),
release(e3), time(e3,past), aspect(e3,simple), voice(e3,active),
pronoun(e4,he),
lsubj(e3,e4)
worm(e5), number(e5,sing), det(e5,the),
lobj(e3,e5),
proposition(e6),
main_event(e6,e3),
lobj(e2,e6)
```

The name information is derived from the Gazetteer lookup stage (where *Morris* is recorded as a male first name), the tense information from the morphological analysis stage, and the grammatical role information from annotations on context-free rules in the grammar. In this case these rules encode that in English sentences which consist of a noun phrase followed by a verb phrase, which in turn consists of a verb in the active voice and a sentential complement, the noun phrase prior to the verb is the subject and the sentence following it is the object. For common nouns and verbs, the lexical root of the word becomes a predicate in the QLF language.

Both noun phrase heads and verb group heads are given unique discourse entity references of the form e_n. This allows modification relations (e.g. of prepositional phrases) or grammatical role information (e.g. subject and object relations) to be captured via binary predicates holding of these entities. In cases where parsing fails to capture all this information (e.g. when only simple noun phrase, verb group, prepositional phrase or relative clause chunks are found and not a spanning parse for the sentence) then partial QLF information can be returned, making the system robust in the face of grammatical incompleteness.

Each sentence in a candidate answer passage is analysed in this fashion. In addition so is the question, using a special question grammar. This grammar produces a QLF for the question in much the same style as above. For example, a question such as *Who released the internet worm in the late 1980s?* would be analysed as:

```
qvar(e1), qattr(e1,name), person(e1),
release(e2), time(e2,past), aspect(e2,simple), voice(e2,active), qcon(e2,verb),
lsubj(e2,e1),
worm(e3), number(e3,sing), det(e3,the),
lobj(e2,e3),
name(e4,'Internet'), qual(e3,e4)
```

Note the use of the special predicate, `qvar` (question variable), to indicate the 'entity' requested by the question. In this case the `qvar` can also be typed because *who* tells us the entity of concern is a person, and we presume (by encoding this in the transduction rules) that the attribute we are seeking here is a name (and not, e.g., a definite description such as *a guy at MIT*). In other cases where the interrogative pronoun is more generic (e.g. *what*) the type of the `qvar` and the attribute sought of it may not be so readily determinable.

2.4.2 Discourse Interpretation of Candidate Answer Texts

Once a text has been parsed and each sentence has been assigned a QLF representation as discussed in the preceding section, the next component of QA-LaSIE, the discourse interpreter, integrates the texts into a discourse model. The discourse model is a specialisation of a semantic net which supplies the system's background domain knowledge. For IE applications, this domain-specific background knowledge assists in extraction tasks, by allowing template slot values to be inferred from it together with information supplied in the text being analyzed. However, for the TREC QA task there is no specific domain, and so this role of the semantic net is not relevant (though a very basic "generic" world model is employed).

The real function of the semantic net in the QA task is to provide a framework for integrating information from multiple sentences in the input. As the QLF representation of each sentence is received by the discourse interpreter, each entity is added as an instance node in the semantic net associated with its type node (the single unary predicate in which it occurs) – e.g. given `worm(e5)`, `e5` is linked to the `worm` node in the net, if it already exists, and to a new node labelled `worm` if not. Added to each such entity node is an attribute-value structure, or property list, containing all the attribute and relational information for this entity (all the binary predicates in which it occurs in the input).

In addition to adding a sentence's QLF to the semantic net in this fashion, one further node is added representing the sentence itself. This sentence entity has a sequence number indicating the sentence's position in the text, and also has an attribute recording the entity numbers of every entity occurring in the text. Thus, the discourse model aims to model not only the content of the discourse, but simple aspects of the discourse structure itself.

After each sentence has been added to the discourse model, the main task of the discourse interpreter commences. This is to determine coreference relations between entities in the current sentence and entities already added to the model from previous sentences in the input. There is not space to detail this algorithm here (see [4]), but in essence it relies upon factors including the semantic type compatibility, attribute compatibility, and textual proximity of potential coreferents. Once a coreference has been established between two entities, the two are merged by replacing all references to the two entity numbers by references to just one of them. However, the surface realisations which initially served as triggers for the creation of each distinct entity node are retained as attributes of the merged entity, and can be used later, e.g. to generate a text string as an answer.

2.4.3 Answer Identification

Given that a discourse model for a candidate answer passage has been constructed as just described, the QLF of the question is added to this model and treated as sentence 0. The coreference procedure is run and as many coreferences as possible are established between entities in the question and those in the passage[2].

In the TREC-8 version of QA-LaSIE this procedure was the primary question answering mechanism: if the `qvar` was resolved with an entity in the text then this entity became the answer; if not, then no answer was proposed. This approach had several major drawbacks. First, it permitted only one answer per question, whereas the QA track allows up to five answers to be proposed. Second, it was very fragile, as coreference tends to be difficult to establish.

Given these weaknesses, the TREC-9 system follows a significantly different approach. Instead of attempting to directly corefer the `qvar` with an entity in the text, entities in the text are scored in a way which attempts to value their likelihood as answers. The best scores are then used to select the answers to be returned from the passage.

The details of this approach are as follows. The discourse model is transversed twice, sentence by sentence:

1. *Sentence Scoring* On the first pass, the sentences are given an integer score. The entities in the question are interpreted as "constraints" and each sentence in the answer text gets one point for each constraint it contains. This has the effect that sentences which contain entities that have been detected as coreferring with entities in the question will be rewarded. Typically this will be sentences which contain named entities mentioned in the question, or sentences which have definite noun phrases or pronouns which have already been resolved (as part of discourse interpretation of the passage).

2. *Entity Scoring* On the second pass, the system looks in each sentence for the best possible answer entity. To be considered a possible answer, an entity must be an object (not an event), and must not be one of the "constraints" from the previous step. If the `qvar` has a tt qattr (see 2.4.1 above), then the entity must also have the specified attribute to be considered a possible answer. The entities in a given sentence are compared to the `qvar` and scored for semantic similarity, property similarity, and for object and event relations.

[2]The standard coreference procedure uses a distance metric to prefer closer to more distant potential coreferences. Clearly this is irrelevant for questions which are not part of the original text. Hence we have switched off the distance-preference heuristic for coreference in this case.

Semantic and property similarity scores are determined as for generic coreferencing. A semantic similarity score between 0 and 1 is computed, depending on how closely semantically related two things are. For instance, if the `qvar` has the type `person`, then an entity that also has type `person` will receive a semantic similarity of 1. In general, the semantic similarity is related to the inverse of the path length that links the two semantic types in the ontology. If the two semantic types are on different branches of the hierarchy, the score is 0.

The property similarity score is also between 0 and 1 and is a measure of how many properties the two instances share in common and how similar the properties are.

The object and event relations scores were motivated by failure analysis on the original system and were tuned through test runs. The object relation score adds 0.25 to an entity's score if it is related to a constraint within the sentence by apposition, a qualifying relationship, or with the prepositions *of* or *in*. So if the question was *Who was the leader of the teamsters?*, and a sentence contained the sequence *... Jimmy Hoffa, Leader of the Teamsters, ...* then the entity corresponding to *Jimmy Hoffa* would get the object relation credit for being apposed to *Leader of the Teamsters*.

The event relations score adds 0.5 to an entity's score if:

(a) there is an event entity in the QLF of the question which is related to the `qvar` by a `lsubj` or `lobj` relation and is not the `be` event (i.e. derived from a copula construction), and

(b) the entity being scored stands in the same relation (`lobj` or `lsubj`) to an event entity of the same type as the `qvar` does. So if the question was, *What was smoked by Sherlock Holmes?* and the answer sentence was *Sherlock Holmes smoked a pipe*, then the entity *a pipe* would get the event relations credit for being in the `lobj` relation to the verb *to smoke*.

This represents a significant weakening of the requirement in our TREC-8 system that the `qvar` had to match with an entity in the answer text which stood in the same relation to its main verb as the `qvar` did with the main verb in the question, as well the main verbs and other complements being compatible. Here a bonus is awarded if this the case; there it was mandatory.

Finally, the entity score is normalized to bring it into the range [0,1]. This is motivated by the idea that if two sentences have equal scores from step 1. above, the entity score should break the tie between the two, but should not increase their scores to be higher than a sentence that had a better score from step 1. Normalizing the score improved performance slightly in tests on the Trec 8 questions.

3. *The Total Score* For every sentence, the "best" answer entity is chosen according to the Entity Scoring as described above. The sentence and entity scores are then added together and normalized by dividing by the number of entities in the question plus 1 The sentence instance is annotated to include the total score, the best entity (if one was found), and the "exact answer". The exact answer will be the `name` of the best entity if one was identified during parsing. Otherwise this property is not asserted.

2.4.4 Answer Output

The answer output procedure gathers the sentence total scores, as described in the preceding section, from each sentence in each of the passages analyzed by QA-LaSIE, sorts them into a single ranking, and outputs answers from the overall five highest scoring sentences.

We submitted four runs to the TREC-9 evaluation, two in the 50-byte category and two in the 250 category. These four runs are explained below:

System	Run	Mean Reciprocal Rank	Correct Answers	Rank in Class
shef-trec8	50	.081	N/A	15/17
okapi-baseline	50	.157	N/A	14/17
shef50ea	50	.329	89/164	4/17
shef50	50	.368	98/164	3/17
shef-trec8	250	.111	N/A	22/24
okapi-baseline	250	.395	N/A	11/24
shef250	250	.490	127/164	4/24
shef250p	250	.506	130/164	4/24

Table 1: Development Results on TREC-8 Questions

shef50ea This is the "exact answer" run, submitted in the 50-byte category. If a high scoring sentence was annotated with a `trec9_exact_answer` attribute then this is assumed to be the answer. If there is no "exact answer", then the code looks for a `trec9_answer_entity` and outputs the longest realization of that entity as the answer. If there is no "answer entity", which can happen occasionally, then a default string is output. In all cases, the string is trimmed to 50 bytes if necessary, by trimming characters from the left hand side.

shef50 For this run, the system looks for the first occurrence of the `trec9_answer_entity` in the sentence and then outputs 50 bytes of the sentence centered around that entity. The 50-bytes will never go outside of the answer sentence (if the first occurrence is the first word, then the 50 bytes will be the first 50 bytes of the sentence, and so on). If the sentence is shorter than 50 bytes, then the full sentence is output as the answer. If there is no answer entity, the middle 50 bytes are output.

shef250 Same as shef50, but up to 250-bytes or the full sentence is output (whichever is shorter).

shef250p For this run, the answer for shef250 is computed, then the answer is padded to 250 bytes if necessary by adding characters from the file to both ends, going outside the confines of the sentence if necessary.

3 Results

In the next two sections we describe results obtained from the system, first during development, and then in the TREC-9 test run.

3.1 Development Results

Our development setup consisted of the 200 TREC-8 questions, the TREC-9 document collection (a superset of the TREC-8 collection), and Perl patterns for identifying correct answers for the TREC-8 questions in proposed answer strings, made available by Ellen Voorhees following the TREC-8 evaluation. These resources allowed us to modify our system, re-run against the 198 TREC-8 questions and score our results in a tight modify-evaluate loop.

One initial baseline experiment was to see if QA-LaSIE was actually adding value over a naive approach that simply used Okapi passage retrieval with a maximum passage length of one paragraph and then trimmed this paragraph to 50 or 250 bytes. Taking the top 5 one paragraph passages for each query in the development set and trimming them to the central 50 or 250 bytes led to MRR scores of 0.157 for the 50 byte responses and .395 for the 250 byte responses. This totally naive approach would have placed 14-th of 17 in the TREC-8 50-byte system ranking and

System	Run	Mean Reciprocal Rank		% Correct Answers in Top 5		Rank in
		Strict	Lenient	Strict	Lenient	Class
shef50ea	50	.159	.172	23.6	25.7	28/35
shef50	50	.206	.217	31.1	32.1	21/35
median (of 35)	50	.227				
mean (of 35)	50	.220	.227	31.0	32.2	
shef250	250	.330	.343	48.5	49.4	28/43
shef250p	250	.345	.357	50.9	51.3	23/43
median (of 43)	250	.349				
mean (of 43)	250	.351	.363	49.0	50.5	

Table 2: TREC-9 Results

joint 11-th of 24 in the 250-byte system ranking. In both cases these results were considerably higher than our own entries in TREC-8.

Thus, we started with a sobering baseline to contend with. However, following development of the new approach described above in section 2.4.3 and numerous experiments with various parameter settings we arrived at the best development results presented in Table 1. For comparison the Sheffield results from TREC-8 and the Okapi baseline experiment results are also included in this table.

3.2 Final Evaluation Results

Mean reciprocal rank scores for the four Sheffield runs are shown in Table 2, for both lenient and strict scorings. We have also computed the percentage of questions for which a correct answer was present in the top 5 answers returned by system. The final column shows the system's rank in the various answer categories, with respect to the number of participating systems (this is calculated with respect to the strict scoring of mean reciprocal rank). Also included are the median and mean scores for systems participating in each category. From these figures it can be seen that in both 50 and 250 byte categories the better Sheffield system is close to, but just slightly below, the median and mean (shef50 is third highest below median, and shef250p second highest below median).

4 Discussion

At this point we do not have the information to allow us to apportion faults between Okapi and QA-LaSIE. In training on the TREC-8 questions (but against the TREC-9 document collection) Okapi was returning answer-containing passages for about 83% of the questions. On this basis the best QA-LaSIE mean reciprocal rank scores obtained in development were around .37 for the 50-byte runs and just over .50 for 250-byte runs, as presented above in Table 1.

Thus the TREC-9 test results represent a significant drop with respect to training results. Nevertheless, with respect to our best TREC-8 MRR results (.081 for the 50-byte run, .111 for the 250-byte run), these figures represent a very significant improvement, especially given that the question set is significantly larger and the questions are "real", as opposed to what were artificially created back-formulations in many cases in TREC-8. And, they validate the central hypothesis of our TREC-9 work that we should abandon our previous rigid approach in which answer text entities either met constraints imposed by the question or did not, in favour of a looser approach which scored potential answer entities in terms of various factors which suggested that they might be an answer.

Acknowledgements

The authors would like to thank Hideo Joho, Department of Information Studies, University of Sheffield, for help with installing and parameterising Okapi.

References

[1] E. Brill. A simple rule-based part-of-speech tagger. In *Proceeding of the Third Conference on Applied Natural Language Processing*, pages 152–155, Trento, Italy, 1992.

[2] J.P Callan, W.B. Croft, and S.M. Harding. The INQUERY retrieval system. In *Proceedings of the 3rd International Conference on Database and Expert System Applications*, pages 78–83, 1992.

[3] R. Gaizauskas, H. Cunningham, Y. Wilks, P. Rodgers, and K. Humphreys. GATE – an Environment to Support Research and Development in Natural Language Engineering. In *Proceedings of the 8th IEEE International Conference on Tools with Artificial Intelligence (ICTAI-96)*, pages 58–66, Toulouse, France, October 1996.

[4] R. Gaizauskas and K. Humphreys. Quantitative Evaluation of Coreference Algorithms in an Information Extraction System. In S. Botley and T. McEnery, editors, *Discourse Anaphora and Anaphor Resolution*. John Benjamins, London, 2000.

[5] K. Humphreys, R. Gaizauskas, S. Azzam, C Huyck, B. Mitchell, H. Cunningham, and Y. Wilks. Description of the LaSIE-II system as used for MUC-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, 1998.

[6] K. Humphreys, R. Gaizauskas, M. Hepple, and M. Sanderson. University of Sheffield TREC-8 Q & A System. In *NIST Special Publication 500-246: The Eighth Text REtrieval Conference (TREC 8)*, 1999.

[7] S. Robertson and K. Sparck-Jones. Relevance weighting of search terms. *Journal of the American Society for Information Sciences*, 27(3):129–146, 1976.

[8] S. Robertson and S. Walker. Okapi/Keenbow at TREC-8. In *NIST Special Publication 500-246: The Eighth Text REtrieval Conference (TREC 8)*, 1999.