The University of Sheffield's TREC 2005 Q&A Experiments

Robert Gaizauskas, Mark A. Greenwood, Henk Harkema,
Mark Hepple, Horacio Saggion and Atheesh Sanka
{r.gaizauskas,m.greenwood,h.harkema,m.hepple,saggion}@dcs.shef.ac.uk

Department of Computer Science
University of Sheffield, UK

1 Introduction

Our entries in the TREC 2005 QA evaluation continue the experiments carried out as part of TREC 2004 and hence we report work on multiple approaches to both the main and document ranking tasks. As well as continuing with our separate approaches we have concentrated common tasks and resources to allow for better more principled comparison of our approaches.

2 Common Resources

Our entries to the TREC QA evaluation in 2003 [Gaizauskas et al., 2003] and 2004 [Gaizauskas et al., 2004] were produced using two independently implemented QA systems which relied on independently developed resources which while containing much that was duplicated did not fully overlap. This made it difficult and unfair to compare the performance of our two systems. Many of these resource have now been combined to provide a single basic knowledge store from which the approaches can draw information. This work is documented in the following sections.

2.1 AQUAINT Indexing

Lucene: Two of our three runs use Lucene[1] to index and access the AQUAINT collection. In previous years the approaches to index creation were not consistent and so later processing was in some cases carried out over different document sets making comparisons between our approaches problematic at best.

For our runs which use Lucene to access the AQUAINT collection a single document processing and indexing approach has now been adopted. Each document is split into separate paragraphs using the embedded SGML paragraph tags. All remaining SGML tags are then removed and each paragraph is added to the Lucene index along with the unique document ID and associated date.

MadCow: To improve are results using the MadCow search engine it we implemented a semantic filter which would discard documents which did not contain an entity of the same type as the expected answer type. Determining the expected answer type is a two stage process.

In the first stage the question is parsed using SUPPLE Gaizauskas et al. [2005] which contains specific question grammars which embed in the semantics of the question a unary question predicate referred to as the `qvar`. The second stage then involved mapping the `qvar` to one of the known semantic entity types using mostly hand-crafted lookup tables. The result is a semantic type which represents the expected answer type (EAT) of the question. Table 1 contains some example questions along with their `qvar` and EAT.

Consider the question *"What city is Disneyland in?"*, SUPPLE determines that the `qvar` is `city`. The `qvar` is then mapped through lookup tables which allows us to determine that the correct EAT for a `qvar` of `city` is Location:locType=city. If the question grammar is unable to find a question variable then the EAT for the question is `null`.

The EAT can be classified as either general or specific. General EATs are those that specify just a high-level semantic type, for example Organization, Location, Person. Specific EATs are those which specify a lower-level semantic type or an attribute of a high-level type. For example Measurement:kind=number, Organization:orgType=company, Location:locType=city, and Person:gender=male are all specific EATs.

[1] `http://lucene.apache.org/`

	Question	qvar	Exact Answer Type
1894	*How far is it from Earth to Mars?*	measure	Measurement:kind=number
1898	*What city is Disneyland in?*	city	Location:locType=city
1909	*What business was the source of John D. Rockefeller's fortune?*	business	Organization
1924	*When was the first hair dryer made?*	date	Date:kind=date
1935	*What color is the top stripe on the U.S. flag?*	color	Color
1950	*Who created the literary character Phineas Fogg?*	person	Person
13.2	*What actor is used as Jar Jar Binks voice?*	actor	Person:gender=male

Table 1: Questions with qvar predicates and EAT annotation

2.2 Target and Question Processing

Both our approaches to QA assume that each question is asked and answered in isolation. This is different to the current TREC approach of a set of questions related to a given target.

In 2004 we used two simple approaches to deal with merging targets and questions: pronoun replacement and appending the target to the question. Neither method was ideal and both failed to produce acceptable results. For the 2005 evaluation we adopted a single shared approach based on both pronominal and nominal coreference resolution.

For example consider the seven questions for target 75, Merck & Co., and the processed questions which result:

	Original	Modified
75.1	*Where is the company headquartered?*	*Where is Merck & Co. headquartered?*
75.2	*What does the company make?*	*What does Merck & Co. make?*
75.3	*What is their symbol on the New York Stock Exchange?*	*What is Merck & Co.'s symbol on the New York Stock Exchange?*
75.4	*What is the company's web address?*	*What is Merck & Co.'s web address?*
75.5	*Name companies that are business competitors.*	*Name companies that are business competitors.*
75.6	*Who was a chairman of the company in 1996?*	*Who was a chairman of Merck & Co. in 1996?*
75.7	*Name products manufactured by Merck.*	*Name products manufactured by Merck & Co.*

Note that this approach does not always result in a independent question, for example question 75.5 cannot be answered without reference to the target. In cases such as these the target is simply appended to the question to enable relevant documents to be located. It should be clear, however, that in the other questions the target has been successfully inserted into the question including the addition of possessives where necessary.

In this years test set 40 of the 455 factoid and list questions could not be modified to insert the target. This has consequences during retrieval and answering of the question. The insertion failed mainly because the reference to the target was made by a nominal expression or an ellipsis instead of a pronominal expression ("the first flight" for space shuttles or "the center" for Berkman Center for Internet and Society). Of the remaining 405 questions whilst the target was inserted in the question an a few questions this resulted in badly formed or misleading questions. For example question 70.4 was *"What was the affiliation of the plane?"* for the target *"Plane clips cable wires in Italian resort"* for which our approach produced the question *"What was the affiliation of Plane clips cable wires in Italian resort?"*.

2.3 Semantic Entity Detection and Normalization

These merged resources include gazetteer lists and semantic entity recognisers which together allow us to recognise a large number of distinct entity types in free text. This both extends our ability to recognise semantic entities and provides a solid foundation upon which our multiple strategies can be built.

Whilst this work provides a solid foundation for our two QA systems it does not address the problem of there being multiple ways of representing identical pieces of information. The answers to many questions can be represented in many ways and as most QA systems rely at least in part on the frequency of occurrence of competing candidate answers being able to accurately compare candidate answers is important. To this end all dates and numbers were normalised to a standard format. Dates are all converted to a standard numerical format including resolving partial or descriptive dates (such as *today* or *tomorrow*) against the date of the newswire article. Numbers, both isolated and within measurements, are converted to a plain numeric form, i.e. 3000, 3,000, and three thousand are all represented as 3000.

3 Approaches to Answering Factoid and List Questions

3.1 The Shallow Multi-Strategy Approach

Originally introduced as a baseline system for comparison with our main entry in TREC 2003 [Gaizauskas et al., 2003], our shallow multi-strategy approach (SMS) has continued to be improved and is now no longer considered a baseline system. The systems was described in some detail by Gaizauskas et al. [2004] and so we will concentrate just on detailing the main

modifications to the system.

Expanding the Question Hierarchy Using WordNet: Whilst expanding the answer type hierarchy using WordNet proved useful in our TREC 2004 experiments a number of problems did arise. The main issue being that some entries in WordNet, which may appear in questions, should not be used directly to find answers. For example words such as researchers, soldiers, chemists, etc... should not be used directly but should instead be linked back to the Person type within the answer hierarchy. For the current evaluation the WordNet expansion has been tightly integrated with the question hierarchy to enable this mapping which should increase the performance of the approach.

Just Guess the Answers: As the TREC guidelines state that all list questions are known to have answers within the AQUAINT collection those systems which cannot find an answer and are therefore forced to return a dummy response penalise themselves and would be better to simply guess a number of answers. This is because given the evaluation metric there is no difference between returning a single wrong answer or 100 wrong answers.

When our approach fails to find an answer to a list question it guesses answers by assuming that correct answers will occur frequently in relevant documents and that they will be fully contained within noun phrases. Twenty hopefully relevant documents are retrieved and all noun chunks are extracted from them using a version[2] of the Ramshaw and Marcus [1995] base NP chunker.

The noun phrases are then clustered by assuming that two noun phrases are equivalent if the non-stopwords in one are all present in the other. The longest phrase is then used to represent the cluster. These clusters are ranked using a scoring function (the same as that used to rank the answers to factoid questions). Given that a unique answer a to question q has been seen C_a times by the answer extraction component within the retrieved documents the most likely of which occurred in sentence s then the answer is scored using the equation:

$$score(a, q, s) = C_a * \frac{|q \cap s|}{|q|} \tag{1}$$

This scoring function takes into account the fact that it is more likely that a correct answer will not only appear frequently in relevant documents but will also come from a sentence which contains many (if not all) the question words.

If less than ten clusters are found then all are returned as answers to the list question otherwise the first ten chunks are returned along with any others which have a score above 0.08 (chosen by empirical testing over questions from previous TREC evaluations).

3.2 Matching on Logical Forms

QA-LaSIE performs partial syntactic and semantic analysis of questions and candidate answer bearing documents and then performs matching over the derived logical form representation. The system has been described in detail in past TREC proceedings (see, e.g. Greenwood et al. [2002]) and here we will only describe modifications carried out since it last participated in TREC 2004 [Gaizauskas et al., 2004].

Parsing with Semantic Entities: This year we made use of SUPPLE [Gaizauskas et al., 2005], a freely-available, open source natural language parsing system, implemented in Prolog[3]. Entities identified by the semantic entity detection and normalization procedure documented in Section 2.3 are passed to the parser via a mapping process. The entities we considered this year were: Building, Color, Date, Email, Location, Measurement, Money, Organization, Person, and Quote.

These semantic entities are mapped into noun phrases with specific semantics. As an illustration, the expression “23 inches” is mapped into a noun phrase with the following semantics:

```
measurement(e1), count(e1,23),
measurement_type(e1,distance), name(e1,'23 inches')
```

Answer Ranking: The answer scoring mechanism this year uses document ranking in addition to the score we used in previous years [Gaizauskas et al., 2004]. Document rank information is used when two answers have the same sore, the answer found in a document with lower rank – thus more relevant, is proposed first.

[2] We use the Java re-implementation available from `http://www.dcs.shef.ac.uk/~mark/phd/software/`

[3] `http://nlp.shef.ac.uk/research/supple`

4 Approaches to Answering 'Other' Questions

4.1 The Bare Target + Filter + Reduce Approach

This system, introduced and described in Gaizauskas et al. [2004], was used almost unchanged from the system evaluated in TREC 2004 – the only changes being minor bug fixes.

This system assumes that each nugget can be fully contained within a single sentence and so sentences are selected from the corpus only if they contain the target as it appears in the question; no use of coreference was made to increase the number of matching sentences. Each sentence was retained if it did not overlap more than 70% with any sentence already in the definition. The process stopped either when there were no more sentences to process or the definition had reached 4000 characters in length. This approach while effective still results in much repetition within the resulting definitions.

In an attempt to remove more of the redundant sentences from the definitions a second filtering step was introduced. This second filter works by calculating the sum of the percentage overlap of increasingly longer n-grams. The n-grams considered range from length 1 (a single token) to length s which is the length of the shortest of the two sentences being compared. From limited testing a cutoff level of 50 was determined with pairs of sentences having a score above this being deemed equivalent. To increase the number of nuggets returned, rather than reduce the amount of text, the system was updated to create initially a definition of up to 5000 characters. The second filter is then applied and the resulting definition is trimmed to the first x sentences that produce a definition of 4000 characters.

While filtering the sentences allows the system to remove some of the redundancy from the generated queries, it is clear that returning whole sentences still results in a large amount of redundant text being included in the definition. Rather than attempting to extract the salient details from the sentences we attempted to determine a number of phrases and clauses which while being redundant could also be removed from the sentence without affecting either the meaning or flow of the text. To allow the system to answer questions in real-time we only attempt to find redundant words or phrases using shallow methods which do not require intensive processing. This rules out detection of redundant phrases which would require full syntactic or semantic parsing to identify. A number of sources were consulted for possible ways in which redundant phrases could be both identified and safely removed [Dunlavy et al., 2003, Purdue University Online Writing Lab, 2004]. The phrases were removed from the sentences before any filtering is applied as previous work in summarization has shown this to be the most effective point at which to remove redundant clauses [Conroy et al., 2004].

The words and phrases which were deemed redundant and easily removable were: imperative sentences, gerund clauses, leading adverbs, sentence initial expletives, redundant category labels, unnecessary determiners and modifiers, circumlocutions, unnecessary that and which clauses, and noun forms of verbs.

4.2 Target Enrichment + Filter Approach

This approach to answer other questions has changed very little from the same approach we described for the 2004 TREC evaluation[Gaizauskas et al., 2004].

The approach requires each target to be classified as a person ('who' question) or other type of entity ('what' question). Having performed semantic entity detection on the target we used the following procedure to identify the target, its type, and any additional context for the target:

- If the target contains a person's name, then we extract the first 'named' person in the target, considering as context any text to the left and right of the named entity (e.g. for "Abraham in the Old Testament" the target is "Abraham" while "in the Old Testament" is the context). The question is considered to be of type 'who'.
- If the target contains an organization's name, then the first organization is extracted, and the text to the right and left of the target is considered to be the context (e.g., for "ETA in Spain" the target is "ETA" while "in Spain" is the context). The question is considered of type 'what'.
- Otherwise the less discriminative word of the input text is considered the target and any other words are used as context (e.g., for "medical condition shingles" the target is "shingles" and the context is "medial condition"). The question is considered to be of type 'what'.

When searching the web for definitional passages using the approach described by Saggion and Gaizauskas [2004], we use (exact) definitional patterns in the Google query (e.g., "Abraham was a") as well as the identified context (e.g. "in the Old Testament").

The parameters used for the runs `SHEF05MC` and `SHEF05LC` are as follows: the maximum number of characters for an answer was 4000 bytes, the maximum number of nuggets per target was 14, and 1000 documents returned by the document retrieval system (MadCow or Lucene) were used.

5 Results

We submitted the following three runs for evaluation in both the main and document ranking tasks and the performance of these runs is discussed in the following sections.

Run Tag	Retrieved Total	Retrieved Relevant	Known Relevant	Precision	Recall	% Coverage At Rank 20
shef05lmg	789	155	1575	0.196	0.098	70
SHEF05LC	883	186	1575	0.211	0.118	82
SHEF05MC	937	216	1575	0.231	0.137	78
MadCow	1000	219	1575	0.219	0.139	76

Table 2: Summary of results from our three document ranking entries.

Run Tag	Factoid	List	Other	Combined
shef05lmg	0.202	0.076	0.160	0.165
SHEF05LC	0.110	0.035	0.158	0.103
SHEF05MC	0.116	0.039	0.172	0.114

Table 3: Summary of results from our three main task entries.

shef05lmg This run answers factoid and list questions using the SMS approach of Section 3.1 and the bare target, filter, and reduce approach of Section 4.1 to answer other questions. Documents are retrieved from the AQUAINT collection using Lucene.

SHEF05LC This run uses the logical form matching approach of Section 3.2 to answer factoid and list questions along with the target enrichment and filter approach of Section 4.2 to answer other questions. Documents are retrieved from the AQUAINT collection using Lucene.

SHEF05MC This run is identical to SHEF05LC apart from the fact that it retrieves filtered documents from the AQUAINT collection using MadCow.

All three runs made use of the semantic entity detection and normalization of Section 2.3 and the target/question processing of Section 2.2. And all three runs used just the top twenty documents retrieved by their IR approach.

5.1 Document Ranking Task

Before reporting the results of our three entries in the document ranking task it should be noted that whilst both shef05lmg and SHEF05LC runs were produced using Lucene the document ranking evaluations will differ. Unlikely the logical form matching approach the SMS approach analyses each question to see if it can be answered before retrieving any documents. For those questions which could not be answered no documents were retrieved and a single dummy doc ID was returned[4]. This lowers the document ranking score for shef05lmg without affect the ability to answer the questions, i.e. the difference in document ranking scores has no affect on later processing.

The full results[5] for our three document ranking runs can be seen in Table 2.

Interestingly Table 2 shows that precision and recall are not useful for comparing document retrieval runs for QA as the coverage results (where coverage is the percentage of questions for which at least one answer bearing document was retrieved [Roberts and Gaizauskas, 2004]) show that SHEF05LC is capable of answering more questions than SHEF05MC (41 compared to 40) yet has lower precision and recall. This leads us to stress the importance of choosing the correct evaluation metric and to suggest that coverage (and answer redundancy) be more widely adopted.

Also contained in Table 2 is an evaluation of MadCow without filtering. This shows that performing filtering increases the coverage of the retrieved documents (76% to 78%) which it is hoped will improve the end-to-end performance of the QA approach.

5.2 Main Task

Table 3 shows the results of our three entries at the factoid, list, and other questions as well as the combined per-series score. Only brief analysis of the three runs has currently been performed:

shef05lmg: There were 8 targets for which shef05lmg was unable to answer any of the questions (i.e. the series score was 0) although no analysis has yet been undertaken to see if there is a pattern in these failings. For 15 of the other questions

[4] We simply returned the first document in the APW section of the collection namely APW19980601.0003.

[5] The results for SHEF05MC differ slightly from the official results. On rare occasions MadCow allowed more than 20 docs to be returned for a question. These documents were not used in later processing and hence inflate the document ranking scores while obscuring the actual data used in later processing.

whilst at least one nuggets was found no vital nuggets were found and as such the score for these questions was zero even though useful information had been found. This clearly effects the score of this run for both the questions own score and the per-series score.

SHEF05LC: There were 16 targets for which SHEF05LC was unable to answer any of the questions and 10 other questions for which nuggets were found but no vital nuggets were returned giving a score of 0. Again no analysis of these failures has yet been conducted.

SEHF05MC: There were 16 targets for which SHEF05MC was unable to answer any of the questions (no comparison with the 16 targets for which SHEF05LC was unable to answer any questions has yet been carried out) and 5 other questions for which nuggets were found but no vital nuggets were returned giving a score of 0. Again no analysis of these failures has yet been conducted.

6 Discussion

Much more analysis of results needs to be carried out before firm conclusions can be drawn. However, certain observations are worth making even now:

- Whilst the document ranking task showed that using Lucene gave better coverage (i.e. more questions could be answered) using MadCow actually resulted in higher scores across all three question types. This clearly shows that performing document retrieval for use within question answering systems is a complex topic which requires further detailed investigation.
- Both our approaches to answering other questions performed worse than expected (approximately half the score obtained in the TREC 2004 evaluations [Gaizauskas et al., 2004]. We imagine that this is due to the inclusion of more event targets which were complex than we expected.
- Having standardised many tasks and resources common to our approaches allowing us to fairly compare approaches it seems that the SMS approach consistently outperforms the logical form matching approach.

Acknowledgements

Our thanks to the UK Engineering and Physical Sciences Research Council for funding this research through their studentships programme and under research grant GR/R91465/01.

References

John M. Conroy, Judith D. Schlesinger, John Goldstein, and Dianne P. O'Leary. Left-Brain/Right-Brain Multi-Document Summarization. In *Proceedings of the Document Understanding Conference (DUC 2004)*, 2004.

Daniel M. Dunlavy, John M. Conroy, Judith D. Schlesinger, Sarah A. Goodman, Mary Ellen Okurowski, Dianne P. O'Leary, and Hans van Halteren. Performance of a Three-Stage System for Multi-Document Summarization. In *Proceedings of the Document Understanding Conference (DUC 2003)*, 2003.

R. Gaizauskas, M. Hepple, H. Saggion, and M. Greenwood. SUPPLE: A Practical Parser for Natural Language Engineering Applications. In *International Workshop on Parsing Technologies*, 2005.

Robert Gaizauskas, Mark A. Greenwood, Mark Hepple, Ian Roberts, and Horacio Saggion. The University of Sheffield's TREC 2004 Q&A Experiments. In *Proceedings of the 13th Text REtrieval Conference*, 2004.

Robert Gaizauskas, Mark A. Greenwood, Mark Hepple, Ian Roberts, Horacio Saggion, and Matthew Sargaison. The University of Sheffield's TREC 2003 Q&A Experiments. In *Proceedings of the 12th Text REtrieval Conference*, 2003.

Mark A. Greenwood, Ian Roberts, and Robert Gaizauskas. The University of Sheffield TREC 2002 Q&A System. In *Proceedings of the 11th Text REtrieval Conference*, 2002.

Purdue University Online Writing Lab. Conciseness: Methods of Eliminating Wordiness. `http://owl.english.purdue.edu/handouts/print/general/gl_concise.html` [July 2004], 2004.

Lance Ramshaw and Mitchell Marcus. Text Chunking Using Transformation-Based Learning. In *Proceedings of the Third ACL Workshop on Very Large Corpora*, June 1995.

Ian Roberts and Robert Gaizauskas. Evaluating Passage Retrieval Approaches for Question Answering. In *Proceedings of 26th European Conference on Information Retrieval*, 2004.

Horacio Saggion and Robert Gaizauskas. Mining on-line sources for definition knowledge. In *Proceedings of FLAIRS 2004*, Florida, USA, 2004. AAAI.