Experiments in Passage Selection and Answer Identification for Question Answering

Horacio Saggion and Robert Gaizauskas

Department of Computer Science
University of Sheffield
211 Portobello Street
Sheffield - S1 4DP
England - United Kingdom
{saggion,robertg}@dcs.shef.ac.uk

Abstract. Question Answering (QA) aims at providing users with short text units that answer specific, well-formed natural language questions. A two stage architecture is widely adopted for this task consisting of a document retrieval step followed by an answer extraction step. In such an approach two main problems need to be addressed to reduce the search space: better selecting answer bearing passages in the document retrieval step and better pinpointing answers in the answer extraction step. We investigate the effect of word-based and linguistic-based features for the identification of answer-bearing sentences and answer candidates in a QA system and show that both play a significant role.

1 Introduction

Finding textual answers to open domain questions in huge text collections is a challenging problem [Hirschman and Gaizauskas, 2001]. Unlike Information Retrieval (IR) which aims at providing documents satisfying users information needs expressed in the form of a query, Question Answering (QA) aims at providing users with, usually short, text units that answer specific, well-formed natural language questions.

The Text REtrieval Conferences (TREC) [Voorhees, 2002] have clearly defined question types, data, and evaluation methods to assess the performance of QA systems.

In the work reported here we have focused solely on factoid questions which require single facts as answers. The evaluation metric defined by TREC for factoid questions is answer-accuracy: the percent of correct answers found by the system. We have developed a QA system which is composed of an information retrieval (IR) engine followed by an answer extraction (AE) component. This approach is widely adopted by TREC participants. However it has the disadvantage of bounding the performance of the QA system by that of the IR engine. Recent studies in QA [Gaizauskas et al., 2004] have shown that one has to descend deep in the ranked list of documents in order to find passages containing the answer[1]. For example, using NIST Z-PRISE system one has to descend to rank 1000 in order to have 87% coverage[2] over the TREC 2003 factoid questions. At the same time, as the AE component is given lower rank documents to consider more answer hypotheses appear, thus decreasing answer-accuracy. One way of obtaining answer bearing passages at higher ranks is by applying re-ranking methods to the IR output to maximise coverage at lower ranks. Another alternative, which we investigate in this work, is to train classifiers for passage identification so that passages unlikely to contain the answer can be identified and discarded as early as possible (or the classifier's outcome probability used for re-ranking purposes).

Based on previous reported success with the use of boolean search for QA [Harabagiu et al., 2000], we have developed an in-house boolean search engine that performs word indexing at the sentence level (see Section 3) that we used in the experiments described here. For the AE component (detailed in Section 4), we have adopted a NLP-based approach where questions and candidate passages are transformed into semantic representations by a process of robust, partial parsing. Each entity in a candidate passage is scored according to (i) its semantic proximity to the expected answer type (EAT), (ii) the number of linguistic-based relations involving an entity from the question that appear in the candidate passage, and (iii) the similarity between the question and the passage where the candidate comes from. The entity with the highest score is proposed as the answer to the question. For example, question 2219 from TREC/QA *"How tall is Al Pacino?"* asks for a measurement EAT. A suitable candidate passage from AQUAINT to answer this question is document NYT19990811.0068 where the following sentence is found:

> *"Other celebs in their stocking feet: Danny DeVito, 5 feet; Spike Lee, 5 feet 5 inches; Al Pacino, 5 feet 6 inches; and Martin Sheen, 5 feet 7 inches."*

Note that there are many candidate answers in this passage. However, because *"Al Pacino"* (a question entity) and *"5 feet 6 inches"* are in a relation of apposition in the passage, the latter should be considered a more likely answer than other competing candidates: this is the approach taken by our extraction component.

In spite of its linguistic appeal, the value of the linguistic-based features used by our extraction component had yet, prior to the work reported here, to be quantitatively assessed. In this work we try to remedy this situation by addressing two related issues: (i) the identification of answer bearing passages returned by the IR engine as a way of reducing noise; and (ii) the identification of answers in answer bearing passages to boost answer-accuracy.

[1] We use the term "high ranked" documents to refer to documents that are judged more relevant to a query than documents that are "low ranked", though the numeric rank associated with a high ranked document (e.g. 1) is lower than the rank associated with a low ranked document (e.g. 1000).

[2] *Coverage at rank k* is the proportion of questions for which a correct answer can be found within the top *k* retrieved documents [Roberts and Gaizauskas, 2004].

We see both tasks as classification problems and adopt a supervised machine learning (ML) research methodology, making use of the WEKA software [Witten and Eibe, 2000]. In order to carry out experimentation we have created a data-set of answer-bearing sentences, non-answer bearing sentences, and answers for a subset of TREC 2003 factoid questions. In the experiments reported below, we study the effect of word-based and linguistic-based features derived from question/passage analysis on the process of identifying answer bearing passages and answers.

More generally, and from an AI perspective, our work tries to assess the question of to what extent linguistic analysis helps in the QA enterprise.

In the rest of the paper we introduce in detail our IR and AE components. Then, we describe the data set, experiments and results on passage and answer identification. Finally, we close with conclusions and suggestions for further research.

2 Related Work

Question Answering has been part of the artificial intelligence agenda for a long time [Woods, 1973]. However, fuelled by the Internet, the huge volume of on-line texts, and the establishment of the TREC/QA track [Voorhees, 2002], Question Answering is nowadays receiving unprecedented attention from research laboratories and commercial companies. In a traditional document retrieval/answer extraction architecture two main problems need to be addressed: document filtering and answer pinpointing. In order to obtain good answer bearing passages as early as possible, a two-pass strategy can be used which will first extract a large number of candidate passages and then re-rank them by using learned ranking functions [Usunier et al., 2004]. Such an approach has been shown to increase coverage with respect to the unranked IR output. One technique that has been applied to combat the noise produced by the IR system is filtering of answer candidates by type checking the expected answer type. Because named entity recognisers of coarse-grained types of named entities (e.g., dates, places, people, organisations) tend to be quite accurate, checking the type of the candidate answer has a positive impact for certain types of question [Schlobach et al., 2004]. However, and in spite of recent advances in the development of fine-grained question type hierarchies [Hovy et al., 2001], open domain question answering may need additional techniques for answer filtering.

3 Boolean Information Retrieval

In the experiments described in this paper we have made use of an in-house boolean retrieval system which performs word indexing at the sentence level (see [Gaizauskas et al., 2003] for a description of the boolean search engine). A basic boolean query language was implemented which supports queries of arbitrarily deeply nested conjunctions and disjunctions of search terms (words);

negation is not supported at this point as there has not appeared to be a need for it. For boolean retrieval engines the task of formulating queries to retrieve documents relevant to answering some question is non-trivial. A query formed as a conjunction of the words in the question (omitting stoplist words, perhaps) will commonly be too restrictive, returning few or no documents, whereas a query that is a disjunction of the same words will commonly retrieve too many. The best approach for formulating boolean retrieval queries is an open research topic. However, we have recently carried out extensive experimentation and identified a strategy for question analysis and sentence retrieval [Saggion et al., 2004]. This strategy is an iterative process which starts with an initial query and then modifies it until the required number of sentences have been returned or no further refinements to the query are possible.

The strategy for document retrieval achieved 62.15% coverage at rank 200 (on TREC/QA 2003 factoid data) which compares unfavourably to the 80.4% coverage at the same rank for Z-PRISE. However, while at rank 200 our retrieval system will return on average 137 sentences per question, Z-PRISE will return around 4600, broadening considerably the search space for the AE component.

4 Answer Extraction

Answer extraction receives as input a set of candidate passages and returns one answer per question (or the constant NIL if an answer cannot be found). Depending on the IR retrieval component, passages can be full-documents, paragraphs, or as with the IR approach previously described, sentences matching the query.

AE is a pipeline of freely available components which include named entity recognition, POS-tagging, parsing, and discourse interpretation. The system first carries out partial, robust syntactic and semantic analysis of the passages returned by the search engine and of the question, transducing them both into a predicate-argument or quasi-logical form (QLF) representation (see Figure 1). Note the analysis may well be only partially correct - e.g. the EAT in the representation shown in Figure 1 is not the logical object of the verb *to travel*.

In this representation the predicates are, for the most part, either the unary predicates formed from the morphological roots of nominal (e.g. `submarine`) or verbal (e.g. `travel`) forms in the text or binary predicates from a closed set of grammatical relations (e.g. `lobj` for the verb logical object, `lsubj` for the verb logical subject) or of prepositions (e.g. `in`, `after`) or the special binary predicate `name` to identify the name of a named entity. Identifiers (e_n) are created for each entity in the representation.

In this step, the EAT is determined (e.g., `measure`) and depending on the question, a special attribute (`qattr`) is created which indicates the attribute-value to be output from the answer entity. For example in the case of a measurement, the value to extract is an attribute `count` that should be attached to the answer.

Given these sentence level "semantic" representations of candidate answer-bearing passages and of the question, a discourse interpretation step then creates

1937: *How fast can a nuclear submarine travel? QLF:* travel(e2), submarine(e1), lsubj(e2,e1), lobj(e2,e3), adj(e1,nuclear), qvar(e3), qattr(e3,count), measure(e3)

Fig. 1. Parser output for question 1937. `qvar` represents the sought answer.

a discourse model of each retrieved passage by running a coreference algorithm against the semantic representation of successive sentences in the passage, in order to unify them with the discourse model built for the passage so far. This results in multiple references to the same entity across the passage being merged into a single unified instance. Next, coreference is computed again between the QLF of the question and the discourse model of the passage, in order to unify common references.

In this model, possible answer entities are identified and scored as follows. First each sentence in each passage is given a score based on counting matches of entity types (unary predicates) between the sentence QLF and the question QLF. Next each entity from a passage not so matched with an entity in the question (and hence remaining a possible answer) gets a preliminary score according to (1) its semantic proximity to the EAT using WordNet and (2) whether or not it stands in a relation R to some other entity in the sentence in which it occurs which is itself matched with an entity in the question which stands in relation R to the sought entity (e.g. an entity in a candidate answer passage which is the subject of a verb that matches a verb in the question whose subject is the sought entity will have its score boosted). An overall score is computed for each entity as a function of its preliminary score and the score of the sentence in which it occurs. The final answer to the question is the entity with the highest score.

5 Experiments

One of the main goals of this work is to better understand the contribution of various features in use in our system for the passage selection and answer identification tasks.

5.1 Q&A Data

The data used in the experiments reported here is that used in the TREC 2003 QA track. The text collection used is the AQUAINT corpus consisting of approximately one million documents drawn from three newswire sources for the period 1998-2000 (about 3.2 gigabytes of text). The TREC 2003 question set consists of a subset of 150 factoid questions from the collection. In order to support automated evaluation, NIST produces regular expression patterns for each question which match strings containing the answer. In addition to the regular patterns, NIST also provides the document ids where correct answers can be found. In order to identify answer bearing sentences (ABS) from AQUAINT we relied on the documents judged as correct by NIST analysts: each sentence in an answer bearing document is considered an ABS if it matches a question pattern and an additional manual check to verify that the sentence does answer the question.

For example, consider question 2293 *"How many times a day do observant muslims pray?"*. It has as possible correct answers the strings provided by NIST *"five"* and *"at least five"*. The following two sentences were extracted from answer bearing documents identified by NIST:

> (S1) *But at work, it takes about* **five** *minutes counting the time it takes to get to a prayer area and wash their hands if necessary.*

> (S2) *As devout Muslims, they consider it their duty to pray* **at least five** *times a day, one of those times while they are at work.*

both (S1) and (S2) contain an instance of an answer string. However only (S2) is an ABS because it allows one to infer an answer to the question.

In order to identify non answer bearing sentences (NABS), we relied on our boolean search engine. For each question we pulled out from AQUAINT at most 100 sentences following the strategy previously described. Sentences which are not ABS were considered NABS. This strategy has been designed so that the instances used for the experiment simulate a natural QA setting where most of the sentences returned by the IR engine will be NABS. The distribution of sentences in the data set is rather skewed with only 25% of the cases being ABS. Each question and sentence was analysed by the AE parser and transformed into a logical representation. Logical forms and linguistic information such as words, word lemmas, and POS categories are available for feature computation. The full process of linguistic analysis is automatic and as a consequence imperfect.

5.2 Features for Passage Identification

In the experiments reported here, we study the effect of word-based and linguistic features as potential sources of information for ABS identification.

During experimentation we fixed the following word-based features (words and lemmas are normalised to lowercase and stop words are removed): (W1) number of common lemmas between question and sentence, and (W2) number of common words between question and sentence. In addition to these two, four more features represent the size of the respective sets of question/sentence lemmas/words. This set of features which we denote WBF is too simple to differentiate cases which are extremely different. Consider for example question 2384 from TREC/QA: *"What is the population of Canberra?"* and two candidate sentences

> *(S3) "Canberra with a population of 306,400"*

and

> *(S4) "The prospect of an exodus of refugees from a population of 210 million causes alarm from Canberra to Bangkok".*

Both sentences match all nonstop question words and lemmas, thus they are identical with respect to W1 and W2. Note, however, that (S3) contains a relation between *"Canberra"* and *"population"* which is not present in sentence (S4). A relation between *"Canberra"* and *"population"* also occurs in the question, thus making (S3) more likely to contain an answer. It is intuitive to think that the presence in a sentence of entities together with relations which also appear in the question should be used to estimate ABS likelihood.

2392: *When was the Red Cross founded? QLF:* qvar(e1), qattr(e1,name), date(e1), found(e2) lsubj(e2,e3), name(e3,'Red Cross'), in(e2,e1)
F1: $date(X)$
F2: $date(X)$
F3: $found(X)$
F4: $name(X,' RedCross')$
F5: $found(X) \wedge date(Y)$
F6: VOID
F7: $found(X) \wedge name(Y,' RedCross')$
F8: VOID

Fig. 2. Sets for feature computation.

Therefore, for each question we compute four linguistic-based sets: (F1) the expected answer type set of those predicates in the QLF possibly representing the answer; (F2) the predicates in the QLF derived from nouns (objects); (F3) the predicates in the QLF derived from verbs (events); and (F4) the instances of the binary predicate `name` (the named entities). These sets are used as the basis for computation of sets of pairs of predicates (linguistically motivated bigrams) indicating that a relation exists between: (F5) an object and an event; (F6) two objects; (F7) a name and an event; and finally (F8) a name and an object. These F1-F8 sets are used to compute eight features for each sentence representing the number of elements in each set matching the sentence QLF. For sets of bigrams, it is checked whether or not a relation exists between the two components of the bigram in the QLF. In Figure 2 we show how sets F1-F8 are computed for question 2392. An ABS for question 2392 such as:

1863 - International Committee of the Red Cross is founded in Geneva.

having QLF:

```
date(e1), name(e1,1863), name(e3,'Geneva'), organization(e11),
name(e11,'International Comitee'), of(e11,e12), name(e12,'Red Cross'),
found(e10), in(e10,e13), lobj(e10,e11)
```

will have features instantiated as follows F1=1, F2=1, F3=1, F4=1, F5=0 (because `date` and `found` are not related in the sentence), F6=VOID, F7=0 (because e12 is not related to the event `found`), and F8=VOID. Additionally, the size of the respective sets is computed for each question and used as a feature. We refer to this complete feature set as LBF.

We have made use of different classifiers from the WEKA toolkit in order to identify whether the data described with word-based features (WBF) or linguistic-based features (LBF) helps the learner in this binary classification task. We use the *ZeroR* classifier as a baseline against which we compare the more informed methods. Its strategy is to classify all sentences according to the most frequent category: NABS in our case.

5.3 Results

We have experimented with several algorithms and obtained statistical improvement over the baseline. Table 1 shows the performance (in terms of *%correct*) of J48, a particular WEKA implementation of the C4.5 decision tree algorithm which was one of the most accurate classifiers. It outperforms the baseline (at a 99% confidence level) using either of the two feature sets[3]. Interestingly, there is no statistical difference in classification accuracy between J48 using either WBF ($J48_{WBF}$) or LBF ($J48_{LBF}$). In order to verify if the two sets of features could be used together in order to improve classification accuracy we have specified a pos-hoc classifier (Comb) which uses the outcome and posterior probability of the J48 classifiers. Given a sentence representation S, Comb computes $J48_{WBF}(S)$ and $J48_{LBF}(S)$ and their respective outcome probabilities $P_{WBF}(S)$ and $P_{LBF}(S)$. Comb classifies S as $J48_{WBF}(S)$ if $P_{WBF}(S) \geq P_{LBF}(S)$ otherwise it classifies S as $J48_{LBF}(S)$. This algorithm outperforms significantly (confidence level %99) both WBF and LBF (See Table 1).

Classifier	ABS	NABS	ALL
Baseline	0 (0%)	3507 (100%)	3507 (74%)
$J48_{WBF}$	708 (57%)	3477 (99%)	4185 (88%)
$J48_{LBF}$	824 (66%)	3349 (96%)	4173 (88%)
Comb	861 (69%)	3432 (98%)	4293 (90%)

Table 1. Experimental results for answer bearing sentence identification. Column ABS indicates the number of ABS instances correctly classified. Column NABS indicates the number of NABS correctly classified. Column ALL indicates the number of instances correctly classified.

[3] Apart from differences in *%correct*, statistically significant differences are observed in *precision*, *recall*, and *F-measure*

6 Features for Answer Identification

For the experiments described here, we need to identify for each entity in an answer-bearing sentence whether or not it is an answer. We carry out this task in two steps, firstly identifying the offsets of the answer string provided by NIST, and then extracting from the QLF the entity id (e_k) that is realised within those offsets.

We use a set of six features derived from the linguistic intuitions used in the AE component. In order to compute them we first create for each question the following sets of lambda expressions: (L1) the expected answer types set of those predicated in the QLF representing the answer; (L2) the set of expected attributes from the answer (e.g. `qattr`); (L3) the predicate the sought entity is the logical subject of; (L4) the predicate the sought entity is the logical object of; (L5) the set of predicates the sought entity is related to by a relation different from subject or object; (L6) the set of predicates occurring in the question (excluding the EAT). Figure 3 shows the required sets for feature computation for an example question. Given a sentence and an answer candidate, each lambda expression (L1-L6) is instantiated with the candidate hypothesis and matched against the sentence logical form. The corresponding feature will be true if the instantiated expression matches the QLF, otherwise it will be false. Consider again the following answer bearing sentence for question 2392:

1863 - International Committee of the Red Cross is founded in Geneva.

and its partial QLF:

```
...found(e10), in(e10,e13), name(e3,'Geneva'),
date(e1), name(e1,1863)...
```

Figure 4 shows how the features are instantiated for entities $e1$ (an answer) and $e13$ (a non-answer). For example in order to test (L5) with entity $e1$ the goal $found(Y) \wedge in(Y, e1)$ (e.g., $e1$ is attached to the event `found` by relation `in`) will be matched against the sentence QLF. The class of the entity will be 'answer' or 'non-answer' according to the procedure described above.

For this task again, the distribution of answer/non-answers is very skewed with only 6% of positive instances. This again leaves very little space for classification improvement. We have experimented with different classifiers from the WEKA toolkit and have compared them with *ZeroR* classifier which classifies each entity as a non-answer.

6.1 Results

The best performing algorithms were again decision trees (J48). The classification performance of this algorithm is shown in Table 2. Two $J48$ instances are presented, one uses the full set of features ($J48_{ALL}$) while the other uses only features L1 and L2 ($J48_{L1;L2}$). Both configurations outperform significantly the baseline at a 99% confidence level. Algorithm $J48_{ALL}$ has a modest absolute improvement in classification accuracy over $J48_{L1;L2}$ which is statistically significant at 99% confidence level.

2392: *When was the Red Cross founded? QLF:* qvar(e1), qattr(e1,name), date(e1), found(e2) lsubj(e2,e3), name(e3,'Red Cross'), in(e2,e1)
L1: $\lambda X.date(X)$
L2: $\lambda X.name(X,Y)$
L3: VOID
L4: VOID
L5: $\lambda X.found(Y) \wedge in(Y,X)$
L6: $\lambda X.found(X) \vee \lambda X.name(X,'RedCross')$

Fig. 3. Features for answer identification (question 2392).

Features	L1	L2	L3	L4	L5	L6
e1	true	true	VOID	VOID	false	false
e13	false	false	VOID	VOID	true	true

Fig. 4. Features computed for entities $e1$ and $e13$ from an ABS from document APW19981023.1385

6.2 Discussion

In our experiments with sentence classification both word-based features and linguistic-based features combine to outperform either of the feature sets individually. It should be noted that in further experiments with the learning environment, we have come to the conclusion that the EAT is not the only feature responsible for classification accuracy.

Our experiments on answer classification provide interesting insights into the role of the linguistic features used in our QA system. Our results seem to indicate that features from question analysis help in the answer classification task and that the EAT is not the only factor in answer identification. This result however should be interpreted with caution: the results of answer classification indicate that from the 150 questions in the set:

– 2 questions (1%) would be incorrectly answered because no true answer is identified as such, instead two false positives are output;
– 14 questions (9%) could be correctly and unambiguously answered using the classifier because only true positives and true negatives are output;
– 16 questions (11%) would need an additional disambiguation step (probably taking into account answer redundancy) because the classifier outputs both true positives and false positives; and finally
– 118 questions (79%) would remain unanswered because all instances were classified as non-answers (true negatives and false negatives)

These results, however considerable, are rather modest. This can be attributed to the following facts. On the one hand, the data used in our experi-

Classifier	ANSWER	NON-ANSWER	ALL
Baseline	0 (0%)	6351 (100%)	6351 (93%)
$J48_{ALL}$	111 (24.77%)	6296 (99.16%)	6390 (94.26%)
$J48_{L1;L2}$	91 (20.31%)	6300 (99.22%)	6390 (94.02%)

Table 2. Accuracy results for answer identification.

ments, because automatically produced is far from correct, making sophisticated-relational features rather sparse. On the other hand, the features studied here are derived from the 'superficial' forms obtained through question analysis, thus ignoring the mismatch problem between relations in the question and similar relations in the answer-bearing sentence. The latter could be addressed by incorporating a sophisticated process of paraphrase identification [Lin and Pantel, 2001].

7 Conclusion and Future Work

A number of valuable results have emerged from this work. First, our experiments on answer-bearing sentence classification show that a number of easy-to-compute word-based features combined with linguistically-motivated ones help in the classification task. However, the success of the classification should be tested in a working environment and the contribution of each individual feature better assessed. Because of the unbalanced characteristic of the data set, it seems better to use the sentence classifiers for sentence ranking, postponing decisions until answer extraction takes place.

Second, in experiments on answer identification, linguistic-features used by our system seem to make a contribution in answer pinpointing. This result should be carefully examined, however. Our experiments depend strongly on a process that accurately identifies answer-bearing sentences and so leaves little space for ambiguity. In order to assess the impact of the parsing process in the accuracy of classification, we intend to either assess the accuracy or our parser or replicate these experiments using an evaluated system. We believe that answer classification could be used here again to rank entities according to their answer likelihood.

In future work, the resources produced in this work will be assessed using TREC/QA evaluation metrics by incorporating the classifiers into our QA system and measuring end-to-end performance.

The approaches we have studied here rely on natural intuitions about how answers to questions should behave in what the linguistic structure of questions and answer bearing passages is concern. However with the ever increasing availability of QA data, an inductive approach could help identify new features which are not directly associated with our linguistic intuitions. This approach is part of our research agenda.

References

[Gaizauskas et al., 2003] Gaizauskas, R., Greenwood, M., Hepple, M., Roberts, I., Saggion, H., and Sargaison, M. (2003). The University of Sheffield's TREC 2003 Q&A Experiments. In *Proceedings of the 12th Text REtrieval Conference.*

[Gaizauskas et al., 2004] Gaizauskas, R., Hepple, M., and Greenwood, M., editors (2004). *IR4QA: Information Retrieval for Question Answering*, SIGIR Workshops.

[Harabagiu et al., 2000] Harabagiu, S., Moldovan, D., Paşca, M., Mihalcea, R., Surdeanu, M., Bunescu, R., Gîrju, R., Rus, V., and Morărescu, P. (2000). FALCON: Boosting Knowledge for Answer Engines. In *Proceedings of the 9th Text REtrieval Conference.*

[Hirschman and Gaizauskas, 2001] Hirschman, L. and Gaizauskas, R. (2001). Natural Language Question Answering: The View From Here. *Natural Language Engineering*, 7(4).

[Hovy et al., 2001] Hovy, E., Geber, L., Hermjakob, U., Lin, C.-Y., and Ravichandran, D. (2001). Question Answering in Webclopedia. In *Proceedings of the 9th Text REtrieval Conference).*

[Lin and Pantel, 2001] Lin, D. and Pantel, P. (2001). Discovery of Inference Rules for Question Answering. *Natural Language Engineering*, 7(4).

[Roberts and Gaizauskas, 2004] Roberts, I. and Gaizauskas, R. (2004). Evaluating passage retrieval approaches for question answering. In *Advances in Information Retrieval: Proceedings of the 26th European Conference on Information Retrieval (ECIR04)*, number 2997 in LNCS, pages 72–84, Sunderland. Springer.

[Saggion et al., 2004] Saggion, H., Gaizauskas, R., Hepple, M., Roberts, I., and Greenwood, M. A. (2004). Exploring the Performance of Boolean Retrieval Strategies for Open Domain Question Answering. In *Proceedings of the SIGIR Workshop on Information Retrieval for Question Answering (IR4QA)*, Sheffield, UK.

[Schlobach et al., 2004] Schlobach, S., Olsthoorn, M., and de Rijke, M. (2004). Type Checking in Open-Domain Question Answering. In *Proceedings of EACI*, pages 398–402.

[Usunier et al., 2004] Usunier, N., Amini, M., and Gallinari, P. (2004). Boosting Weak Ranking Functions to Enhance Pssage Retrieval for Question Answering. In *Proceedings of the SIGIR Workshop on Information Retrieval for Question Answering*, pages 53–58.

[Voorhees, 2002] Voorhees, E. M. (2002). Overview of the TREC 2002 Question Answering Track. In *Proceedings of the 11th Text REtrieval Conference.*

[Witten and Eibe, 2000] Witten, I. and Eibe, F. (2000). *Data Mining: Practical machine learning tools with Java implementations.* Morgan Kaufmann, San Francisco.

[Woods, 1973] Woods, W. (1973). Progress in Natural Language Understanding - An Application to Lunar Geology. In *AFIPS Conference Proceedings*, volume 42, pages 441–450.

This article was processed using the LaTeX macro package with LLNCS style