The *Alyssa* System at TREC QA 2007: Do We Need Blog06?

Dan Shen Michael Wiegand Andreas Merkel
Stefan Kazalski Sabine Hunsicker Jochen L. Leidner* Dietrich Klakow
Spoken Language Systems
Saarland University
D-66125 Saarbrücken, Germany
lsv_trec_qa@lsv.uni-saarland.de

Abstract

We describe the participation of the Saarland University LSV group in the DARPA/NIST TREC 2007 Q&A track with the Alyssa system, using an approach that combines cascaded language-model based information retrieval (LMIR) with data-driven learning methods for answer extraction and ranking.

To test the robustness of this approach that was previously proven on news data also across document collections of varying levels of subjectivity, we test the hypothesis that the answer accuracy over factoid questions does not decrease significantly if blog data is added.

Our results show that on the contrary, the method remains competitive on larger datasets with mixed content, such as the union of the AQUAINT 2 (news) and BLOG 06 (blog) corpora (Macdonald and Ounis, 2006). We also present evaluation results on an unofficial set of questions manually generated from BLOG 06 documents, which were created at LSV.

1 Introduction

This paper describes the revised version of *Alyssa*, the open-domain question answering (Q&A) system developed at Saarland University, the TREC 2007 Q&A track. In general, it follows the statistically-inspired system we already presented at last year's TREC (Shen et al., 2006). Again, this includes a more principled, i.e. an information theoretically well-founded approach. But, in this year's experiments we could benefit from our design choice which resulted in a software construction that served as an experimental platform in the longer term. Therefore, we experimented with some new modules that could be easily included to the existing *Alyssa* system.

We scored well in this year's Q&A track placing fourth in the overall evaluation. With regard to factoid question, we placed third and on definition questions we even placed second.

We report our experiments for the TREC 2007 question answering track, where we used a cascade of LM-based document retrieval, LM-based sentence extraction, maximum entropy based answer extraction followed by web- and knowledge-based answer validation.

In addition, we present the experiments with our new modules which include the processing of the new BLOG 06 data, data-driven query construction, a question type specific document fetching, semantic answer extraction and answer validation.

The remainder of this paper is structured as follows: Section 2 describes the architecture of our system. In Section 3, we describe our experiments with the revised modules used for TREC 2007, and Section 4 compares the results of our different runs. Finally, Section 5 concludes the paper with a summary and possible future work.

2 System Overview

For this year's question answering track, our existing system Alyssa – which already successfully participated in last year's competition – has not only been adapted to this year's task, but also enhanced in various ways.

Figure 1 illustrates the architecture of our current system. The question first undergoes question analysis which is a linguistic analysis compris-

*present affiliation: School of Informatics, University of Edinburgh

ing full and shallow parses and named entity tagging. Along the information of our question typing module, a query is constructed from this analysis. This query is run against document retrieval on the new AQUAINT 2 corpus, the BLOG 06 corpus and passage retrieval on a Wikipedia index. Due to the heavy increase of the amount of data to be indexed for document retrieval, we implemented a two-staged indexing procedure retrieving documents from individual indices first, and then creating a second index from the retrieved documents of the two individual corpora on the fly. Apart from the new corpora to be used in document retrieval, the other most notable change is a dynamic document fetching in which the number of documents to be retrieved depends on the actual question type of a question.

As far as sentence extraction is concerned, we preserved our retrieval approach based on language modeling. The two existing answer extraction modules comprising a surface-based pattern and dependency-based approach have been supplemented by a third component which ranks candidate answer sentences due to their similarity to the question in terms of semantic frame structures from FrameNet.

Answer validation has also undergone some heavy revision. Our previous web-based answer validation using Google has completely been re-designed. In addition to this, we have also added a second validation layer which makes use of various structured databases, such as IMDB, Discogs.com and the CIA World Fact Book.

As far as the different question types are concerned, we mostly focused on improving the performance on factoid questions, thus preserving more or less the modules we used for answering list and definition questions from last year.

3 New Experiments

3.1 Question Typing

For question classification at TREC 2007, we use the same theoretical models as last year (Shen et al., 2006). In this section, we give a short summary how this model looks like and present the dataset we used to re-train this year's language models.

As classification paradigm for the task, we used a Bayes classifier

$$\hat{c} = \arg\max_{c} P(Q|c)P(c) \qquad (1)$$

where Q is the question and c the question type. This model is used because it is known to produce the minimum number of misclassifications if the correct probabilities are known. The probability $P(Q|c)$ can easily be calculated using a language model (LM) trained on all questions of class c. Specifically, we used absolute discounting in a variant known as Kneser-Ney smoothing for bigram language models and a count specific discounting technique for our experiments (Merkel and Klakow, 2007b). The prior $P(c)$ can also be considered as a unigram language model.

For question classification, we use the taxonomy as proposed by (Li and Roth, 2002). It contains 6 coarse and 50 fine grained classes. In our system we used the fine grained classification only because we use specific sub-classes (like DATE, which is a specific sub-class of NUMBER) later on in our sentence retrieval. For this year's question classification, we extended the existing taxonomy by classes described in Table 1. This was done because last year's Q&A tracks showed an increasing number of questions of those types.

Because there is no training data for the types introduced in Table 1, we had to extend the original 5,500 questions provided by the Cognitive Computing Group at University of Illinois at Urbana-Champaign[1] by the new classes. But to get a more reasonable number of training data for those classes, we additionally annotated the TREC 2001-2006 questions. So, the current training set contains about 10% of questions labeled with the new classes.

3.2 Query Construction

We implemented a data-driven method for query construction similar to (Monz, 2007) which differs from our last year's approach (Shen et al., 2006) which is mainly based on simple bag-of-words. The aim of this method is to learn to predict a subset of terms occurring in the question which – when used as query terms for document retrieval – should produce optimal results. (Monz, 2007) proposes *gain*, a measure based on *average precision* that predicts how useful a single term is for a given question. In the question *In which country is Luxor?*, *country* is not expected to appear in the actual answer sentence and should, therefore, not be part of the query. To predict the *gain*, a decision tree is learned which uses a set of syntactic, se-

[1] http://l2r.cs.uiuc.edu/∼cogcomp/Data/QA/QC/

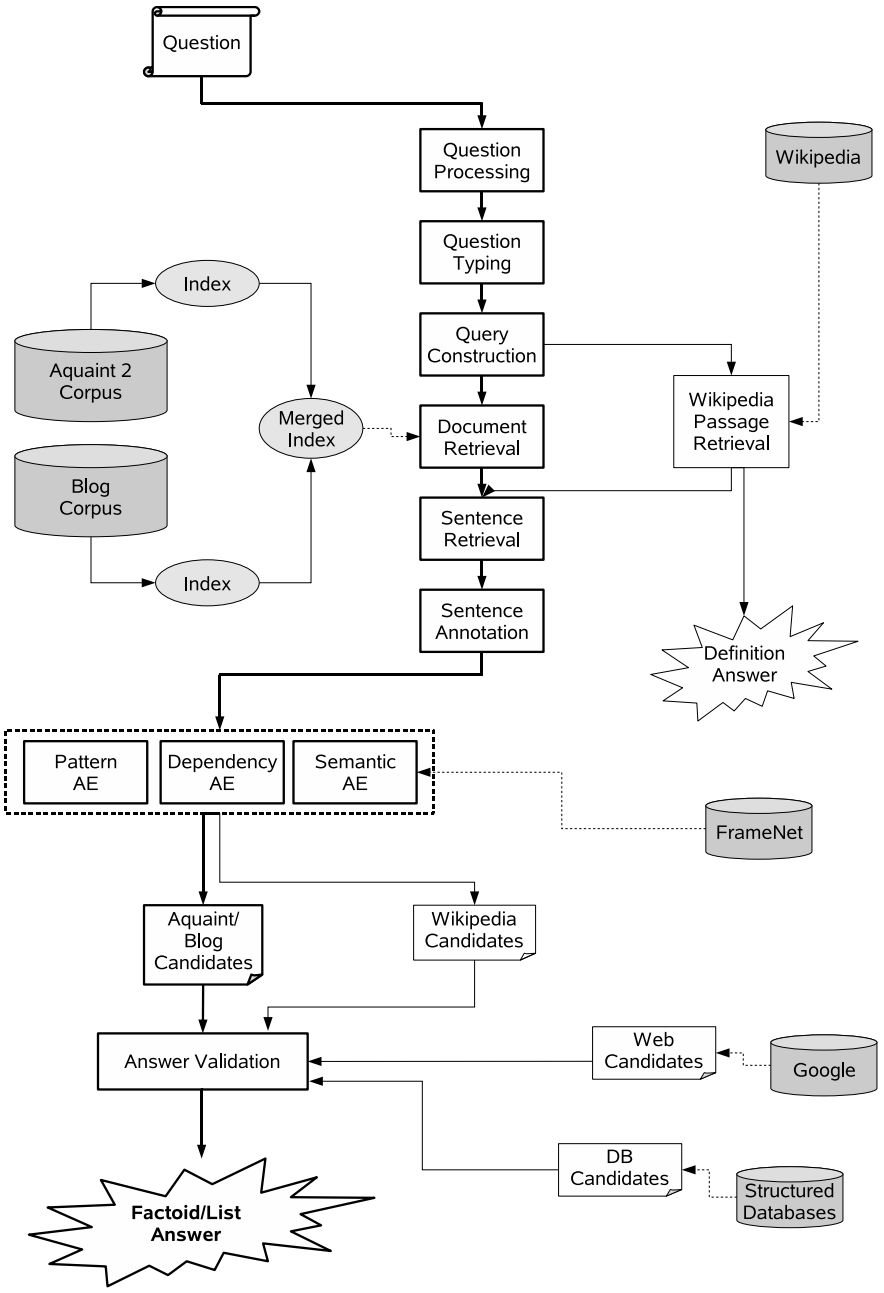

Figure 1: Architecture of *Alyssa*.

New Class	Description
ENTY:cremat:movie	all moving images
ENTY:cremat:books	any printed matter
ENTY:cremat:song	any piece of music
HUM:actor	actor of movies and TV programmes
HUM:author	author of printed matter
HUM:composer_performer	musical performer and/or composer

Table 1: List of the New Question Types Used for TREC 2007 Experiments.

mantic and surface-based features characterizing the individual terms in their respective context.

We adjusted this method to our system by using a slightly different way of compiling the *gain* measure. We tested our implementation by performing a three-fold crossvalidation on the TREC-QA tracks 9, 10 and 11.

Our tests, however, showed that this approach does not work for our system. On each track the method scored significantly lower than our original query construction. Further inspection of the classifier revealed that only $50 - 60\%$ of the potential query terms were classified correctly. Due to the poor performance of the new data-driven query construction, we decided to re-use our original query construction for this year's track.

3.3 Two Level Indexing Strategy

In this section we describe our integration of the BLOG 06 and the AQUAINT 2 corpus. AQUAINT 2 is a newswire corpus similar to AQUAINT. The documents contained in this corpus are proofread, so we can expect the texts to be written in proper English. BLOG 06 is a newly created corpus comprising common blog pages from the web. Therefore, these web sites are not proofread and can contain misspellings, ungrammatical formulations and incorrect punctuation marks. The documents are unprocessed web pages extracted from the web and, thus, include HTML tags and script language fragments.

We carried out three steps to prepare both corpora for document retrieval. In the first step, we extracted the plain text from the BLOG 06 documents by removing HTML tags and script language fragments. For this purpose, we used *CyberNeko Tools for XNI*[2]. We preserved tables of contents and other meta-information that are not part of the actual document. Fortunately, the language model based sentence extraction consistently ignores these text fragments. In the second preprocessing step, sentence boundaries are detected in BLOG 06 and AQUAINT 2 documents using an HMM-based approach. Finally, the documents with annotated sentence boundaries are indexed by *Lemur Toolkit for Language Modeling and Information Retrieval* [3].

A single corpus comprising all documents from BLOG 06 and AQUAINT 2 would result in a huge corpus containing $3,215,171 + 906,777$ documents. Unfortunately, we were not able to use the language modeling module of Lemur for these data sizes. The other document retrieval methods, such as TF-IDF or Okapi, were working on these data but previous experiments on AQUAINT show that the document extraction using language modeling performs much better than TF-IDF or Okapi (Merkel and Klakow, 2007a). So, we decided to split the document retrieval in two stages. In both stages we used language modeling to extract the documents.

1. In the first stage, documents are extracted separately from BLOG 06 and AQUAINT 2. We extract the same amount of documents from both corpora. The size of documents to be extracted is larger in the second stage than in the first. The more documents are extracted during the first stage the better a one-stage document retrieval is approximated.

2. After extracting documents from both corpora we create a new corpus from the extracted documents. This corpus is much smaller than a merged corpus of both corpora.

3. After building the merged corpus the relevant documents are extracted for further processing.

[2] http://people.apache.org/∼andyc/neko

[3] http://www.lemurproject.org

Method	MAP	R-prec	P@10	bPref
Two-Stage	0.2646	0.335	0.1353	0.6549
One-Stage	0.2539	0.2328	0.1342	0.5514
Okapi	0.2249	0.2088	0.1300	0.5106
TF-IDF	0.2052	0.188	0.1069	0.4917

Table 2: Results of Document Retrieval: TREC 2006 Question Set on AQUAINT.

	One-Stage	Two-Stage
MAP	0.2661	0.2646
R-Prec	0.2328	0.335
P@10	0.1342	0.1353
bPref	0.8116	0.6549

Table 3: Results of Document Retrieval: Comparison between One-Stage and Two-Stage Retrieval.

Table 2 shows the results obtained by different retrieval methods. In our experiments we used the question set from TREC 2006 and extracted the documents from AQUAINT. 30 documents were extracted in each run, because our QA System performs best with this number of extracted documents. We obtained the parameter settings for each model from previous experiments (Hussain et al., 2006). We observe that TF-IDF and Okapi perform worse than language modeling in a one-stage document retrieval. Our two-stage document retrieval also performs better on a single corpus than the one-stage document retrieval. Furthermore, the amount of irrelevant documents that are ranked higher than relevant documents decreases in our two-stage document retrieval. This can be explained by the filtering out irrelevant documents in the first stage resulting in a less noisy language model in the second retrieval stage. Overall, the language model based document extraction performs better than traditional document extraction methods like TF-IDF or Okapi. The results in these experiments confirm our previous experiments.

3.4 Dynamic Document Fetching

In our experiments we discovered that slight variations in the number of extracted documents affects the overall performance of our QA System. Using a question type independent document retrieval, we obtain best results when 30 documents are extracted per question. Additionally, we observed that the global change in performance induced by varying the number of retrieved documents does not always correlate with the changes in performance observed on the individual question types. Figure 2 illustrates the dependency of extracted documents, question type and number of correct answers. We can observe, for example, that for the coarse question type HUM extracting 10 documents leads to worse performance than extracting 30 documents. The question type LOC, on the other hand, behaves in the opposite way.

The fine grained question types show similar changes in performance. With the optimal number of documents for each question type we can improve the performance of our system for the TREC 2006 question set on AQUAINT from 83 correct factoid answers to 93 correct answers. This is an increase of 12%. But this optimization would lead to overfitting, so we decided only to change the number of documents for question types with clear maxima and leave other question types that fluctuate unchanged. Thus, we get a document retrieval which improves the performance by 6 questions, an increase of 7%.

The dynamic document extraction could be further improved by including artificial question type dependent terms. For example, a special document extraction for the question type LOC using the artificial term *location* for the presence of an entity of that kind in a candidate answer document increases the performance on these questions by 4 correct answers. Such improvements need, however, further investigation to avoid overfitting.

3.5 Semantic Role-based Answer Extraction

In TREC 2006 (Shen et al., 2006), we developed a maximum entropy-based answer ranking

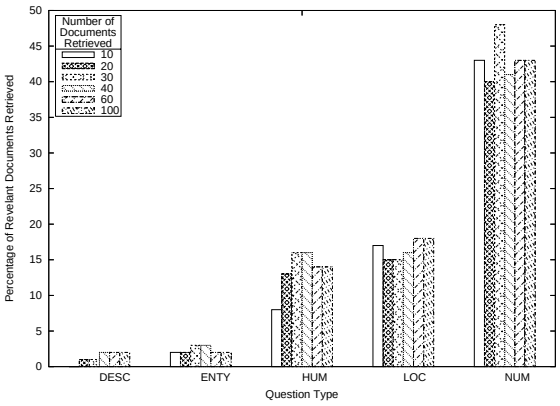

Figure 2: Correct Answers per Question Type and Extracted Documents.

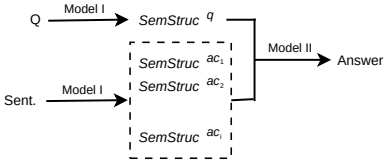

Figure 3: Architecture of Semantic Structure-based Answer Extraction.

module, which mainly captures the evidences of expected answer type matching, surface pattern matching and dependency relation correlation between question and answer sentences. This year, we further incorporated a new answer extraction component (Shen and Lapata, 2007) by capturing evidence of semantic structure matching. Shallow semantic parsing (Carreras and Màrquez, 2005), the automatic identification and labeling of sentential constituents, has recently received much attention. Our work is motivated by examining whether semantic role information is beneficial to question answering. We construct a general framework to exploit semantic role annotations in the FrameNet (Fillmore et al., 2003) paradigm. Once question and answer sentences are normalized to a FrameNet-style representation, the answer candidate whose semantic structure is most similar to the question will be thought most likely to be correct.

Figure 3 shows the architecture of the semantic structure-based component. Semantic structures for questions and sentences are automatically derived using the Semantic Structure Generation Model (Model I). A semantic structure $SemStruc = \langle p, Set(SRA) \rangle$ consists of a predicate p and a set of semantic role assignments $Set(SRA)$. p is a word or phrase evoking a frame F of FrameNet. A semantic role assignment SRA is a ternary structure $\langle w, SR, s \rangle$, consisting of frame element w, its semantic role SR, and score s indicating to what degree SR qualifies as a label for w.

For a question q, we generate a semantic structure $SemStruc^q$. Question words, such as *what*, *who*, *when*, etc., are considered expected answer phrases (EAPs). We require that EAPs are frame elements of $SemStruc^q$. For each candidate ac, we derive its semantic structure $SemStruc^{ac}$ and assume that ac is a frame element of $SemStruc^{ac}$. Question and answer semantic structures are compared using Semantic Structure Matching Model (Model II). We calculate the similarity of all derived pairs $\langle SemStruc^q, SemStruc^{ac} \rangle$ and incorporate them as features of the answer candidates respectively.

For the Semantic Structure Generation Model, We view semantic structure as a bipartite graph. In the graph, nodes in left side are words/phrases of a sentence and nodes in right side are semantic roles of the frame evoked by the identified predicate of the sentence. The weights of the edge connecting the left-side nodes to right-side nodes represent how compatible the phrase and semantic roles are. The initial bipartite graphs are constructed by comparing dependency relation paths attested in the FrameNet annotations and the sentence. Then we search for an optimal role assignment by solving a global optimization problem in a bipartite graph. As a soft-assignment, it allows to assign more than one semantic roles to one phrase, which goes some way towards addressing coverage problems related with FrameNet.

Once the semantic representation is generated, we develop the Semantic Structure Matching Model to measure the similarity between two semantic representations. The precondition of the matching is that predicates of the semantic representations evoke same/related frames. Once it is satisfied, the semantic role matching are further conducted. It is formalized as a graph matching problem. Finally, semantic-structure-matching features of an answer candidate are fired using the matching scores between question and contexts of the answer candidate.

The details of the semantic structure-based AE component are discussed in (Shen and Lapata, 2007). This year, we use the FrameNet V1.3 lexical database. It contains 10,195 predicates grouped into 795 semantic frames and 141,238 annotated sentences. Since FrameNet is still under construction, there are 3,380 predicates with no annotated sentences and 1,175 predicates with less than 5 annotated sentences. For the factoid questions this year (360 questions), there are 83 questions mismatching predicates in FrameNet, such as *Q 229.5: Who supervised the transplant?*. In the rest 277 questions, predicates of 22 questions contain no/less than 5 annotated sentences in FrameNet, such as the predicate *evolve* of the question *Q 224.5: WWE evolved from what earlier organization?*. In principal, there are 255 questions of which semantic structure-based analysis

might have potential contribution.

3.6 Web Validation

The vast amounts of information available on the World Wide Web makes it an attractive resource for answer finding. (Breck et al., 2001; Clarke et al., 2001) state that answer redundancy in an enormous collection of unstructure, flat texts has a strong correlation with answer correctness. The core idea of the approach is that the volume of available web data is large enough to supply the answer to most factual questions multiple times and in multiple contexts varying from complicated and implicit contexts where sophisticated natural language processing is required to simple and explicit contexts where only surface pattern matching may work well.

This year, we have developed a web validation module to further validate top-ranked answer candidates returned by Answer Extraction module. It uses the frequency of the candidates within web data to vote for the most likely answer. We obtain web data by using Google Search Engine. Various queries ranging from loose to tight are generated. For example, for the question *Q 223.3: Where is Merrill Lynch headquartered?*, the following queries including Bag-of-Words (BOW), Noun-Phrase-Chunks (CNK) and Declarative Forms (DEC) are constructed:

- **BOW**: Merrill Lynch headquartered
- **CNK**: "Merrill Lynch" headquartered
- **DEC**: "Merrill Lynch is headquartered"

The declarative form of a question is heuristically constructed. We manually develop about 20 patterns for the transformation. For the first two queries, the top 3 snippets Google returned are the following:

- **Merrill Lynch** rose to prominence on the strength of its brokerage network ... Florida U.S., Global Private Client, Regional **Headquarters** for Latin American ...
- With their help, he also arranged to host 150 teachers and administrators at **Merrill Lynch headquarters** for an allday staff meeting.
- Access **Merrill Lynch headquarter**'s address and subsidiary locations, along with insight related to Merrill Lynch executives, competitors, and operations.

For the the third query, Google returned the snippets:

- Today's **Merrill Lynch is headquartered** in lower Manhattan, not far from its original location on Wall Street.
- **Merrill Lynch is headquartered** in New York.
- **Merrill Lynch is headquartered** in the World Financial Center.

It is obvious that the DEC (tightest) query leads to the most explicit answers than BOW and CNK queries. On the other hand, it is also more probable to fail to get snippets from Google. In our experiment, we assign different weights to occurrences of the answer candidate in snippets evoked by different queries. The weight for BOW, CNK and DEC queries are set to 1:2:5 respectively. For each query, top 50 snippets are used for validation.

As to the counting of answer candidate's frequency, specially for quantity-seeking questions, such as *Q 243.4: How many bombers were killed in the London terror bombing attacks?*, the frequency of the answer candidate, such as *4*, will be calculated by matching the head noun-jointed phrase, such as *4 bombers* in Google snippets. For each question, top 10 answer candidates from Answer Extraction Module are validated.

We evaluated the contribution of the Web Validation module on TREC 2006 factoid questions (403 questions). By Answer Extraction module, 175 questions are correctly answered considering top 10 answer candidates while only 83 questions are correctly answered considering answer candidates on the first rank. We use a Web Validation module to further validate and re-rank the top 10 answer candidates (175 questions should be the upper bound of Web Validation Module). Finally, 122 questions are correctly answered by Web Validation by considering answers on the first rank. It shows that the Web Validation module further significantly improve the performance by 46.9% based on the Answer Extraction module.

3.7 Knowledge-based Postprocessing

The motivation to add a knowledge-based component to our existent answer validation is two-fold: firstly, we observed that there are certain types of recurring questions which cannot be answered by

our existing answer extraction modules. Many errors can be ascribed to the inability of our named-entity tagging to recognize various types. Some prominent types are *movie titles* and *song titles*. In addition to this, knowledge bases might also compensate shortcomings of our surface-based answer extraction. The content of the structured databases we considered and our surface-based extraction are very similar in their expressive power. They both answer questions representing very simple but frequently occurring binary/ternary relations. There are a handful of questions, such as questions asking for *nicknames* or *real names*, which cannot be answered by our current pattern-based component because our pattern set is currently too small.

Secondly, answers extracted from structured knowledge bases might enable picking the correct answer from a ranked list of retrieved snippets generated by our answer extraction which – due to its statistic nature – does not have to be the optimal ranking.

This knowledge-based validation is the last component to be called in our system since it requires the output of various other components. The design of the submodules is described below.

3.7.1 Question Patterns

Each question is matched against a set of manually defined firing patterns. Each firing pattern is associated with some simple binary/ternary relation, such as *x-isMovieStarringActor-y*, where all arguments except one are already instantiated - the missing argument is to be retrieved from some knowledge base, e.g. *x-isMovieStarringActor-y*, {*x=?,y=Christopher Reeves*}. For establishing these relations we only use lexical information and named-entity tagging which have already been processed for each question during question processing. Only those questions for which a pattern fired are processed further. The remaining questions are exempt from this postprocessing.

3.7.2 Querying the Knowledge Bases

The incomplete relation is translated to a query to some knowledge base. Currently we use the following resources:

- International Movie Database (IMDB): The knowledge base cannot only be used for questions asking for movie and television appearances of actors/actresses but also as a source of general biographic data of virtually any famous person (such as *place/date of birth, cause of death, real/full name, marital status etc.*)
- Discogs.com: We use this database for retrieving discographies of musical artists.
- CIA World Fact Book: This popular resource for QA systems contains various factoid information about every single state in the world.

3.7.3 Answer Re-Ranking

If an answer could be successfully retrieved from one of the databases, we, first try to find this text snippet in the ranked list of candidate answers retrieved from the AQUAINT 2 and BLOG 06 corpus – after it has been re-ranked by web-validation. In case of a match, we submit this text snippet along its document id as the final answer.

3.7.4 Backprojection

If an answer could be successfully retrieved from one of the databases but not be found in the output of answer extraction, a backprojection is carried out in order to find a document with the answer snippet in the document collection. One has to make sure that the snippet in the document collection is a supportive answer and not just a coincidental occurrence which appears in a completely different context than the question to be answered. Therefore, we used terms from the original question and the answer snippet as the query for document retrieval. Due to the heavy smoothing in retrieval, however, these rankings still contained incorrect documents on the top ranks. In order to find the best possible document we matched each of the top 10 retrieved documents with the exact answer string along various spelling alternatives. These variations were, in particular, necessary for temporal questions. As far as location and temporal questions were concerned, additional variations varying in specificity were added and ordered from specific to general. For example, if the answer is *23rd November, 1963*, the order would be *23rd November, 1963*, *November, 1963* and *1963*. (We have omitted various alternative formats because of the limited space.)

3.7.5 Evaluation

We evaluated our new knowledge-based validation module on the three previous TREC years.

	Factoid		List	
TREC year	*Standard*	*DB Valid*	*Standard*	*DB Valid*
2004	54	60	74	79
2005	90	103	127	129
2006	81	97	112	147

Table 4: Evaluation of the Structured Database Validation (Correctly Answered Questions).

Exact matches	2 (0.0645)
Inexact matches	0 (0)
Not matched	29 (0.9355)
Total factoids	31 (1)

Table 5: *Without BLOG06*: Evaluation on Run-b (Using only AQUAINT2)

Exact matches	3 (0.0968)
Inexact matches	1 (0.0323)
Not matched	27 (0.8710)
Total factoids	31 (1)

Table 6: *With BLOG06*: Evaluation on Run-c (Using AQUAINT 2 and BLOG06)

We evaluated both on list and factoid questions. The results are displayed in Figure 4. Fortunately, we managed to get improved performance on all TREC years – with respect to both factoid and list questions.

3.8 Comparison between Blog and Aquaint 2 questions

TREC assessors were instructed by NIST to include many questions whose answer was only to be found in the BLOG06 corpus but not in the AQUAINT-2 corpus.[4] This NIST policy was not made public at development time, but we had chosen to create our own internal blog question test set from BLOG06 snippets that can serve as answers. However, as can be seen from Tables 5 and 6, respectively, the questions derived this way are too difficult for the current system.

4 Results

We carried out our TREC experiments on three nodes part of a Linux Beowulf cluster with 2.6 GHz Intel Xeon multi-core CPUs (512 MB RAM each). While the cluster is equipped with a master node featuring a 1 TB RAID system, we only used the nodel-local 300 GB hard disk drives to avoid delays caused by NFS overhead.

Table 7 shows the configuration variations of the three runs we submitted and performance (Accuracy) of our system. The first run retrieved answer candidates from only the AQUAINT corpus without Dynamic Document Fetching (DynDoc) and Knowledge-based Postprocessing (KnowPost), whereas the second run incorporated DynDoc and KnowPost. The third run further added Blog06 Corpus to retrieve answer candidates with the supporting of two-level indexing strategy.

As the numbers show, all of our three runs performed better than the median of participants. `lsv2007b` significantly outperformed `lsv2007a`, which may benefit from the incorporation of DynDoc and KnowPost components. Moreover, it indicates that the DynDoc and KnowPost component consistently work well on both the development data (TREC 2006 questions) and the test data (TREC 2007 questions). `lsv2007c` achieved the best performance among three runs. It slightly outperformed `lsv2007b` by answering 6 more questions. `lsv2007c` placed fourth in the overall evaluation, third in the factoid task and even second in the definition task. Although it is surprising to see that the Blog06 corpus is short of sufficient contribution, it did not harm the overall performance. We further go inside the best run of `lsv2007c`. More surprisingly, none of the first ranked answers (360 answers) of all factoid questions are retrieved from Blog06 Corpus.

Another interesting observation is that our definition QA module performed quite good this year. The performances among the three runs do not differ a lot and they are quite close to the best of participants, although only a basic Wikipedia-based snippet retrieval component was implemented in our system. This may be attributed to the fact that Wikipedia is a valuable external resource for definition questions.

[4] Personal communication, Hoa Trang Dang (2007-09-27)

Run ID	AQUAINT2	BLOG06	DynDoc	KnowPost	Accuracy FACTOID	F-score LIST	F-score OTHER
lsv2007a	+	-	-	-	0.233	**0.101**	0.294
lsv2007b	+	-	+	+	0.272	0.085	0.283
lsv2007c	+	+	+	+	**0.289**	0.099	**0.299**

Table 7: LSV Group Runs and Results Submitted to TREC 2007.

5 Conclusion

We have presented our experiments with an improved version of Alyssa for the TREC 2007 Q&A track. One of the innovations of the improved version of Alyssa is that the number n of n-best retrieved documents is determined dynamically taking into account the question type.

Our main finding is that our approach based on cascaded language model based information retrieval followed by answer extraction using machine-learning does not decrease, but remains competitive, if instead of a news-only corpus like AQUAINT2, an additional corpus of blog posts (BLOG06) is used in a setting where some of the answers occur only in the blogs. We also found that there are actually simple BLOG-specific factoid questions that are notoriously difficult to answer using state of the art Q&A technology.

In future work, a more detailed analysis of the issues specific to Q&A over blog data ought to be studied. Furthermore, blogs lend themselves to answering specific types of questions, such as opinion questions (related to the area of sentiment detection).

Finally, we are planning to compare our successful cascaded LM-based IR technique to various alternatives in more detail.

Acknowledgements

Dan Shen and Michael Wiegand were funded by the German research council DFG through the International Research Training Group "IRTG" between Saarland University and University of Edinburgh.

Andreas Merkel was funded by the BMBF project SmartWeb under contract number 01 IMD01 M.

Jochen Leidner was funded by a Royal Society of Edinburgh Enterprise Fellowship from the Scottish Enterprise.

The authors would like to thank Saeedeh Momtazi for interesting discussions.

References

E. Breck, M. Light, G. S. Mann, E. Riloff, B. Brown, P. Anand, M. Rooth, and M. Thelen. 2001. Looking under the Hood: Tools for Diagnosing Your Question Answering Engine. In *Proceedings of ACL2001 Workshop on Open-Domain Question Answering*.

X. Carreras and L. Màrquez, editors. 2005. *Proceedings of the CoNLL Shared Task: Semantic Role Labelling*.

C. Clarke, G. Cormack, and T. Lynam. 2001. Exploiting Redundancy in Question Answering. In *Proceedings of SIGIR 2001*.

C. Fillmore, C. Johnson, and M. Petruck. 2003. Background to FrameNet. *International Journal of Lexicography*, 16:235–250.

M. Hussain, A. Merkel, and D. Klakow. 2006. Dedicated Backing-Off Distributions for Language Model Based Passage Retrieval. In *Proceedings of Hildesheimer Informatik-Berichte*, Hildesheim, Germany.

X. Li and D. Roth. 2002. Learning Question Classifiers. In *Proceedings of COLING, 2002*, Taipei, Taiwan.

C. Macdonald and I. Ounis. 2006. The TREC Blog06 Collection. Technical Report TR-2006-224. `http://ir.dcs.gla.ac.uk/terrier/publications/macdonald06creating.pdf`.

A. Merkel and D. Klakow. 2007a. Comparing Improved Language Models for Sentence Retrieval in Question Answering. In *Proceedings of CLIN 2007*, Leuven, Belgium.

A. Merkel and D. Klakow. 2007b. Improved Methods of Language Model Based Question Classification. In *Proceedings of Interspeech*, Antwerpen, Belgium.

C. Monz. 2007. Model Tree Learning for Query Term Weighting in Question Answering. In *Proceedings of the ECIR*, Rome, Italy.

D. Shen and M. Lapata. 2007. Using Semantic Role to Improve Question Answering. In *Proceedings of EMNLP 2007*.

D. Shen, J. Leidner, A. Merkel, and D. Klakow. 2006. The Alyssa System at TREC2006: A Statistically-Inspired Question Answering System. In *Proceedings of TREC 2006*.