Utilizing Automatic Coreference Resolution with the Jellyfish Question Answering System

Chris Whidden

October 11, 2007

Abstract

The Dalhousie University NLP group has a Question Answering system named Jellyfish. This is our fourth year of participating in the TREC QA track, and it has brought many improvements. We have added GATE processing and coreference resolution and used it to improve the target marking of Jellyfish. Additionally, a new question relation marking phase was added which identified synonyms and subtypes of question relations.

A thorough analysis of the previous year's question set showed that many errors were due to coreference, passage retrieval, matching, or category errors. The analysis can be extended next year, and can be collaborative with its wiki. This report details improvements to Jellyfish that focus on the coreference and matching errors.

A GATE processing step was added to Jellyfish to provide coreference resolution. The first use of this new information improved the target marking phase of Jellyfish. Other uses of this information may be possible. These improvements focus on the coreference errors identified in the analysis.

A question relation marking phase was also added to Jellyfish. This marks relations that are found in a question involving one or more entities, since they are likely to be found in an answer to the question. Synonyms and subtypes of the question relation are also marked. The new question relation marking provides more information for matching rules to find answers.

The results of the improvements are promising. Despite a large increase in running time, adding coreference resolution to Jellyfish increased accuracy from 7.5% to 9.6%, a 28% improvement. With qrel marking, as well, this rose to an accuracy of 10.6%, a 41.3% improvement. The new target marking worked very well with people, since it marks pronouns and last names. Events and locations were not as well suited.

Future research should look into speeding up the GATE processing and reducing the size of the processed passages. Pre-processing the entire corpus, may be a useful idea. Using the processed passages for other improvements should be explored.

Contents

List of Tables

List of Figures

1 Introduction

1.1 Question Answering

Question Answering is a field of Natural Language Processing that attempts to answer questions stated in a natural language. Natural languages, such as English, are very difficult to process because of ambiguity.

Open-Domain Question Answering aims to extract concise, complete answers from large document collections to questions written in a natural language. Question Answering in this report is restricted to Open-Domain Question Answering.

An important problem that occurs when answering a question is coreference resolution, identifying multiple references to the same entity, Using coreference resolution to identify multiple forms of a question's target can greatly improve the accuracy of question answering. Another important problem is identifying synonyms and alternate forms of a question relation.

1.2 The Jellyfish Question Answering System

The Dalhousie University Natural Language Processing (DNLP) group has a Question Answering system named Jellyfish [7]. This is our fourth year of participating in the The Text REtrieval Conference (TREC) Question Answering track, and we have made several improvements to Jellyfish.

Jellyfish consists of several phases, including passage retrieval, question processing, target marking, question category marking, and matching. It is based on the "mark and match" approach to question answering. This approach uses regular expressions to mark features of a question that are located in passages. Matching regular expressions use those marked features to locate answers [1].

Each news article that Jellyfish examines is one passage. Passages are selected and ranked from the news article corpus, and the top 50 or 100 for each question can be examined by Jellyfish. Passages can be retrieved using information extraction techniques on a question's target, or, for increased accuracy, on the full question.

The question processing phase was previously overhauled. [1] The main idea of this phase is based on finding the syntactic structure of a question. Questions with the same syntactic structure are processed with the same regular expressions and that syntactic structure determines their characteristics. Their differences are be handled by later processing.

Questions can have several characteristics, including a target, a category, one or more entities, question relations, dates, and locations. The target of a question is an entity that the question is generally about: the subject of the question. The category of a question defines what kind of question it is, or the expected answer type. A LOCATION question asks for a location, while a PERSON question asks for a person's name. The entities involved in a question and the relations between them are also important. Often, one of these entities will be the question's target. Additional context information such as the date or location of the answer are also useful.

The target marking phase marks the question's target in each passage retrieved for that question. Jellyfish previously used a simple full text match to mark question targets. We have added a coreference resolution module to our question processing system using the General Architecture for Text Engineering (GATE). Our goal was to explore using more robust processing and advanced techniques to improve Jellyfish. The first test of this coreference resolution module was updating Jellyfish's target marking.

Similar to target marking, question category marking involves marking possible answers to the question. Regular expressions specific to a category are used to mark these.

Another addition to the system is marking relations that are found in questions. These relations, as well as their synonyms are likely to be found in an answer to the question. The relations and synonyms are identified using WordNet, a lexical reference system that is currently very popular in Question Answering [8].

Matching is the phase of Jellyfish that selects answers to the question. Regular expressions containing the marked elements identify those answers. Updated patterns were created that use the marked relations.

1.3 The General Architecture for Text Engineering (GATE)

The General Architecture for Text Engineering (GATE) provides many components for language processing tasks [2]. We used the information extraction and coreference resolution modules to provide coreference resolution for Jellyfish. GATE is simple to use, robust, and did not require a large amount of development effort to implement coreference resolution. However, for the large amount of text required in question answering, using coreference resolution with GATE can be very slow.

GATE uses processing resources (PRs) to process text. Coreference resolution and its required PRs are part of the ANNIE information extraction tools of GATE. The coreference resolution PR is called OrthoMatcher and requires several other PRs to be run on the text first. In order, they are the English Tokeniser which breaks up the document into tokens; the Gazetteer which identifies currency, dates, and other types; the Sentence Splitter which identifies sentences; the POS Tagger which identifies parts of speech; and the NE Transducer which identifies named entities such as people and organizations. We also run the Pronominal Coreferencer which resolves coreference from pronouns. This set of processing resources is specified in a GATE gapp file that GATE can load.

GATE adds information about text in annotations. These annotations can contain features in a feature map. For example, a Person annotation has a "matches" feature that identifies annotations which corefer with it. A GATE PR creates, reads, and edits annotations on the text.

1.4 The Text REtrieval Conference (TREC)

TREC is an annual conference that provides several tracks for the different areas of text retrieval [3]. We are interested in the QA track. TREC question answering is about giving a complete and concise answer to a question in a natural language [4]. Each year, TREC provides a document corpus, questions, and passages for those questions.

Previously, TREC used the ACQUAINT corpus of news articles. This is the first year that the ACQUAINT-2 corpus of news articles has been used. Additionally, this year sees the introduction of a seperate blog corpus.

The main task of the TREC QA track is to answer a set of several hundred questions. Questions are grouped by their target, which is given. Questions can be FACTOID, LIST, or OTHER. FACTOID questions have one answer, LIST questions have multiple answers, and OTHER questions require interesting facts about the target that do not answer previous questions.

Also provided by TREC are the top results from the PRISE search engine, for each target. This allows groups who do not wish to work on passage retrieval to participate in the competition.

There is also often a sub task of the QA track. This year the sub tasks involved complex relation questions, and had an optional interactive component. We chose not to participate in the sub task, so that we could concentrate on the main task.

1.5 Related Work

Several groups used automatic coreference resolution for question answering in the 2006 TREC QA Track, including Fudan University [10], The Language Computer Corporation [5], and MIT CSAIL [6].

Fudan University, similar to many systems, uses coreference resolution to find references to the question target in the question. This does not require the sort of deep processing that this report is about. Jellyfish previously handled this sort of coreference resolution using regular expressions.

The Language Computer Corporation's CHAUCER resolves pronominal and nominal coreference within passages, similar to our system.

Likewise, MIT's START question answering system handles coreference resolution using information about several language features.

2 Analysis of 2006 Questions

To determine the possible improvements that could be made to Jellyfish for this year's set of TREC questions, we performed an in-depth analysis of how Jellyfish handled the factoid questions from last year's set.

The first part of the analysis was a complete Jellyfish run of the 2006 question set using the passages provided by TREC from the PRISE information retrieval engine. The format for this year's question set is unchanged, so this should provide a good comparison. We limited the analysis to the PRISE passage set to focus on the system and not on passage retrieval.

From this complete run we created a script that would print out data about the question as well as possible answers for the question. The possible answers were manually inspected to remove sentences which did not answer the question. In addition, errors in Jellyfish's handling of the question were noted. The analysis was then placed on an internal wiki, to allow collaborative editing and viewing. After the analysis was finished, we determined that many errors were due to coreference issues and that adding a coreference resolution module to Jellyfish could potentially improve the accuracy of our Question Answering system.

2.1 Analysis Format

The analysis follows a simple format for each question. First, there is the question number and the question. This is followed by the question category and answer as determined by Jellyfish. The answer is marked as CORRECT, INCORRECT, or INEXACT. Patterns for a correct answer are next. These are followed by manually entered errors that Jellyfish made in processing the question. The last part of the analysis is a list of sentences that matched the answer patterns from the passages that Jellyfish examined. An example analysis can be found in Figure 1.

The possible processing errors were coreference, category, and other. A coreference error indicates that the target of the question was not properly identified, a category error indicates that an incorrect category has been assigned to the question, and any other errors are identified as other. In Figure 1, the target, Warren Moon, is present in the question as "Moon" but is not identified as the target by Jellyfish. This is a coreference error. Also, Jellyfish has assigned the category LOCATION to a question asking for a position, which is a category error.

The list of potential answers were manually examined. Only sentences that could reasonably be expected to answer the question were retained. The criteria

Figure 1: Example Question Analysis

```
141.1 What position did Moon play in professional football?
   * Category: LOCATION
   * Answer: Cleveland    INCORRECT
   * Patterns:
      * quarterback
   * Processing Errors:
      * Coreference: Moon
      * Category: position
   * Passages:
      * Warren Moon can threaten to retire, but the veteran
      quarterback can't force the Seahawks to budge from their
      position on renegotiating his contract.
```

for retaining a sentence included containing the question target, being of the right category, and supporting other parts of the question such as a date or location. To limit the size of the analysis, only five answer passages were kept for each question.

2.2 Analysis Wiki

To allow for collaborative editing and viewing, the results of the analysis were placed on an internal wiki. A wiki is a web page that allows any user with the appropriate permissions to edit the page. Members of the Dalhousie Natural Language Processing Group working on Jellyfish can use this data to determine how well Jellyfish is currently working, areas to focus interest on, make changes to the analysis, and make comments on the analysis.

2.3 Analysis Results

Results were obtained for 386 factoid questions from the 2006 TREC question set. Since there were 403 factoid questions in total, it is uncertain what happened with the missing 17 questions. Because of this, all results of the analysis will be with respect to those 386 questions.

The analysis showed 140 (36%) coreference errors, 82 (21%) category errors, 334 (86%) found categories, and 2 ($< 1\%$) other errors (strange question processing). Also, 59 (15%) questions did not have any answer in the passages Jellyfish examined.

2.3.1 Common Problems

Aside from the processing errors, there were several common reasons that Jellyfish arrived at the wrong answer: coreference errors, no answer in the provided passages, matching errors, and difficulty identifying members of a category.

Aside from the coreference problems in processing the question, Jellyfish must also deal with coreferences in the article passages. An entity can be referred to by its name, by a pronoun, by its type, or by shorter versions of each. For example, Albert Einstein could be referred to by "Albert Einstein", "He", "the scientist", or even simply "Einstein". Each of these could be found in an answer to a question about that target.

Many answers were not located in the set of passages that was used for this analysis. The PRISE passages provided by TREC are the top documents containing the question target. They do not use the questions to provide more contex, so the answer to a question may sometimes only be found by using custom information retrieval on the full document collection.

Some answers are missed simply because of incorrect matching expressions in Jellyfish. Also, many answers returned by Jellyfish are incorrect because it is difficult to distinguish which answer is correct from multiple passages that contain the question target, a suitable match for the question category, and some supporting context from the question.

Some categories are currently not handled by Jellyfish such as position in the above example. If special cases such as these are not specifically handled, it is difficult to answer the question correctly. However, it is also difficult to specify many categories automatically.

2.3.2 Possible Improvements

Each of the common problems can be improved upon. A coreference resolution module can be added to Jellyfish to determine what references are to the same entity. A document retrieval engine can be used to provide better passages. Matching expressions can be improved. Category matching can also be improved.

Since there are many cases where coreference problems were encountered by Jellyfish, we chose to add a coreference resolution module to Jellyfish. The bulk of this report will focus on that. I will also detail matching improvements using relations found in questions.

However, we are also improving on the other areas. Other DNLP members are experimenting with document retrieval, matching improvements, and category improvements.

3 Jellyfish Improvements

Based on the analysis of the previous year's questions, we decided that adding a coreference resolution module to Jellyfish could lead to a large improvement in accuracy. Of several alternatives, we chose to use the University of Sheffield's GATE to provide coreference resolution [2]. As a first step in using coreference resolution, we updated Jellyfish's target marking component to utilize coreference resolution. One additional area of improvement extends the mark and match approach of Jellyfish to mark relations that are found in questions.

3.1 GATE Processing

The General Architecture for Text Engineering (GATE) provides many components for language processing tasks [2]. We used the information extraction and coreference resolution modules to provide coreference resolution for Jellyfish. GATE is simple to use, robust, and did not require a large amount of development effort to implement coreference resolution. However, for the large amount of text required in question answering, using coreference resolution with GATE can be very slow. We examined several alternatives, including creating our own coreference resolution module, and using the coreference resolution provided with OpenNLP.

3.1.1 Benefits

GATE came highly recommended and has been used by several substantial projects Additionally, GATE is actively maintained and developed since GATE is open source under the Lesser GNU Public License (LGPL). Since GATE provides a coreference resolution module, we did not have to spend a significant amount of time examining typical methods of coreference resolution and implementing our own. Although that could have been a useful research effort, it was not the focus of our project. GATE also provides a graphical user interface for testing and development, although we used it primarily for trying out GATE. Finally, since GATE provides many other language processing components, the effort used to integrate GATE with Jellyfish may also allow us to add other features to Jellyfish more easily and quickly in the future.

3.1.2 Disadvantages

Unfortunately, like any other system, those benefits come with a few tradeoffs. GATE is written in Java and provides Java APIs for its components. Jellyfish is written in Perl, so we were required to develop seperate Java programs to use

GATE and integrate Jellyfish with them. This is not a major problem, but clearly it would be difficult to maintain many interrelated components which were all written in different languages.

Question Answering requires examining a large amount of text, and heavy processing such as coreference resolution can be very computationally expensive. Jellyfish examines 50-100 news articles for each question, and processing them with GATE requires a great deal of time. Our new coreference resolution phase required approximately one day to process the previous year's set of questions with 50 passages per question. However, this year's set of questions with 100 passages per question required almost a week. Passages are processed independently, so we are currently uncertain of the reason for the increase to be more than the two days that we expected. This year's TREC uses a new document collection, so possibly the average article size is larger or some strange construct that GATE takes some time in processing is present more often. Since the PRISE passage set does not differ among the questions for a target, this running time can be improved greatly when using the PRISE passages. We can achieve better performance by using our own information retrieval, however, which requires processing passages for each question.

One other disadvantage of adding GATE processing is the space required for the processed passages. GATE provides two output formats. One format is a specialized GATE format that stores a full graph of annotations. This format increased the size of a passage by an average of 229x. The other format is an inline xml file which stores annotations as tags. This format increased the passage size by an average of 40x. The more compact inline format is easier to use with other programs, but cannot handle cycles and overlapping tags. Since we are not interested in overlapping tags and the inline format gave a space improvement of 5.7x, we chose to use that format. GATE also allows database backed datastores, but there was insufficient time to test one. Both formats store information that is currently redundant to Jellyfish, but may be useful in the future. Removing some of this extraneous information could potentially reduce the space requirement greatly.

3.1.3 Alternatives

The other alternatives that we examined were creating our own coreference resolution module and using the one provided by the OpenNLP project [9]. We quickly decided against creating our own coreference resolution module, due to the time and effort required in development and maintenance. Both of our choices are open source and provide many useful features.

OpenNLP is currently used in the question processing component of Jellyfish [1], so it is already familiar to us and would not have required as much integration work. However, it does not seem to be as actively developed as GATE is, since its newest release was in November of 2005. Additionally, GATE was recommended to us and is used in several larger projects, so we chose to use GATE to provide coreference resolution for Jellyfish.

3.2 Target Marking

As a first test in using coreference resolution with Jellyfish, we chose to update Jellyfish's target marking. The target of a question is provided with the question and is usually a person, organization, or event. Jellyfish marks each location in passages where the target is located. These marked targets are then used in matching expressions to find potential answers to the question.

The previous method of target marking only used an exact match for the target. This meant that any references to the target by last name, type, or via pronouns were ignored by Jellyfish. Our aim was to process passages with GATE and mark coreferences with the target. We split this into two phases, a Coreference Resolution Phase and a Target Marking Phase, so that the coreference data can potentially be used in other areas of Jellyfish.

3.2.1 Coreference Resolution Phase

The Coreference Resolution Phase processes passages with GATE and stores them for later use. Jellyfish stores the passages for each question one per line in a file. We tested running GATE coreference resolution on these files, but this resulted in poor accuracy and performance. Each passage is thus processed individually with GATE and then they are assembled back into a single file per question.

The Coreference Resolution Phase initializes GATE, loads processing resources from a gapp file, process each passage for each question, and outputs the results to a new file for each question. A gapp file uses an xml-based format that specifies the processing resources and their options. They can be manually created or saved from a GATE GUI session.

3.2.2 Target Marking Phase

The Target Marking Phase is our first application of the GATE coreference resolution data. This phase intializes GATE, loads the processing resources from the gapp file, processes the target, loads the GATE processed passages from the previous phase, finds and marks each token that corefers to the target for each question and passage, and outputs each of those passages to a file per question with only the new target markings.

As previously mentioned, targets can be a person, organization, or event. Some targets, especially event targets, may never appear in the passages exactly. Additionally, GATE's coreference resolution only works when an entity is identified as a person or organization. Thus, our Target Marking phase processes targets to identify a substring of the target that contains a person or organization, using GATE.

To find entities that corefer with the processed target, we search GATE's listing of "annotations", GATE's version of tags, for ones that match the processed target exactly. The coreference resolution phase of GATE added a "feature" to the annotation called "matches". This feature contains a colon seperated list of ID numbers which correspond to annotations that corefer with the annotation. We maintain a list of targets, to which these matching annotations are added, unless it already contains them. Exactly matching annotations are also checked against the list. When finished, we have a list of target annotations. The passage is then output with each of the target annotations marked.

3.3 QREL Marking

Jellyfish marks several things to use for matching, including the question target and strings of the question category. One thing it does not mark is question relations. A question relation is a relation involving one or more entities in the question, usually the target. For example, the question "How did Beethoven die?" has the relation die involving Beethoven. The question processing component of Jellyfish identifies these relations, and the matching component includes rules to match them. However, only exact matches of the question relation were previously found. Different tenses and synonyms for "die" such as "died" or "was killed" were ignored.

The QREL Marking component of Jellyfish uses WordNet [8] to identify synonyms of question relations. First the question relation is reduced to a base form by wordnet. Then each word of the question passages is examined to see if it has the same base form, is a synonym, or has a subtype of the same base form. If so, it is marked as a QREL.

The matching expressions were then updated to use these marked QRELs.

4 Results

Adding coreference resolution to Jellyfish showed a marked improvement at the expense of running time. As previously mentioned, coreference resolution is computationally expensive. The new Coreference Resolution phase required approximately one day to process the previous year's set of questions with 50 passages

Question	*Target*	*Coreference*	*Total*	*Marked*	*Correct*	*Accuracy*	*Recall*
141	Warren Moon	YES	11	10	10	100%	91%
153	Alfred Hitchcock	YES	11	9	9	100%	81%
167	the Millennium Wheel	YES	8	0	0	100%	0%
196	Adoption of the Euro	YES	2	1	1	100%	50%
213	Meg Ryan	YES	33	31	31	100%	94%
	Total	YES	65	51	51	100%	63%
	Average	YES	13	10.2	10.2	100%	78%
141	Warren Moon	NO	11	2	2	100%	18%
153	Alfred Hitchcock	NO	11	9	9	100%	81%
167	the Millennium Wheel	NO	8	0	0	100%	0%
196	Adoption of the Euro	NO	2	0	0	100%	0%
213	Meg Ryan	NO	33	7	7	100%	21%
	Total	NO	65	18	18	100%	28%
	Average	YES	13	3.6	3.6	100%	24%

Table 1: Target Marking Comparison

per question However, this year's set of questions with 100 passages per question required almost a week, even though passages are processed independently.

With no improvements, Jellyfish had an accuracy of 7.5% on the previous year's question set. With coreference-based target marking, Jellyfish had an accuracy of 9.6%, a 28% improvement. With qrel marking, Jellyfish had an accuracy of 8.5%, a 13% improvement. With coreference-based target marking and qrel marking, Jellyfish had an accuracy of 10.6%, a 41.3% improvement. Clearly both new features provide a significant improvement to the system.

It is difficult to characterize the recall or performance of the new coreference-based target marking. Since any exact match of the target is found, similar to the previous target marking, as well as coreference matches, the recall must necessarily increase. The accuracy of a target match likely decreases, since errors may be made with the coreference resolution.

To provide some idea of the recall and performance, we randomly chose five passages, each from different questions to manually compare the target marking. Table 1 shows the Question Number, the Question Target, the system type, the total number of coreferences, the number of coreferences marked by a system, the number of correct coreferences marked by a system, the accuracy, and the recall.

From that comparison, we can see that the new target marking works very well with people, such as Warren Moon, Alfred Hitchcock, or Meg Ryan, since it marks pronouns and last names. Events and Locations such as the Millenium Wheel or the Adoption of the Euro are not as well suited. Both systems showed 100% accuracy for this small sample. However, the previous target-marking method had an overall 28% recall and a per-question average of 24% recall, while the new target-marking improved this to an overall 78% recall and a per-question average of 63%. This simple comparison showed a good improvement in recall without a drop in accuracy.

5 Future Work

There are several areas of the new improvements that call for future research, including analyzing this year's question set, speeding up the GATE processing, reducing the size of the processed, passages, pre-processing the entire corpus, prioritizing targets, and using the processed passages for other improvements.

The question analysis should be extended for next year. This will show what should be improved for next year's TREC, and give a better idea of how much the system has improved this year. One area to look into is allowing a question analysis to be updated to reflect the current system. Since a question analysis is created automatically and then updated manually, it can not be updated with the current system's answers easily.

Due to the large amount of time required for the coreference resolution processing, methods should be explored to reduce its running time. Since the newer passages required a disproportionate increase in time compared to their increase in number, one area to examine is the difference between the two passage collections. Perhaps they are much larger on average or more often contain constructs that slow the processing.

Additionally, the large amount of space required to store the processed passages should be examined. Redundant or irrelevant data may be removable. The use of database backed datastores should be examined, as well as the advantages and disadvantages of compression.

One method to greatly increase the online speed and simplicity of the target marking, as well as future improvements that use the processed passages, would be to pre-process the entire data collection. This may, however, require a lot of time and space if the previous two areas are not improved. The time would only be required once, so this would provide a tradeoff in upfront time for later speed.

Currently target marking is a binary process; a target is either marked or not marked. There could potentially be an improvement in accuracy if targets were ranked. For example, exact matches could be ranked higher than partial matches, or targets that are marked more often could be ranked higher. This would add complexity, but may be useful.

The final avenue to explore is utilizing the coreference-processed passages for other improvements. Other than target marking, coreferences could be used to mark possible answers. Other uses for the processed passages should be researched.

6 Conclusions and Recommendations

The Dalhousie University NLP group has a Question Answering system named Jellyfish. This is our fourth year of participating in the TREC QA track, and it has brought many improvements. We have added GATE processing and coreference resolution and used it to improve the target marking of Jellyfish. Additionally, a new question relation marking phase was added which identified synonyms and subtypes of question relations.

A thorough analysis of the previous year's question set is vital to understand possible improvements that can be made to the system. Our analysis can be extended for next year, and can be collaborative with its wiki. The analysis showed that areas to focus on were coreferences, passage retrieval, matching, or category errors.

Adding coreference resolution to Jellyfish showed a marked improvement at the expense of running time. Its accuracy increased from 7.5% to 9.6%, a 28% improvement. With qrel marking, as well, this rose to an accuracy of 10.6%, a 41.3% improvement. Clearly both new features provide a significant improvement to the system.

From our simple comparison of the new target marking to the old, we saw that the new target marking works very well with people, since it marks pronouns and last names. Events and Locations were not as well suited. This simple comparison showed a good improvement in recall without a drop in accuracy. Both systems had 100% accuracy. The new target-marking improved recall from 24% to 63%.

Adding coreference resolution to a question answering system can bring a large improvement. The increase in running time and space can be dramatic, however. Provided that these issues can be mitigated, I recommend adding a coreference resolution phase to question answering systems.

Marking question relations, their synonyms and their subtypes can also improve a question answering system that uses matching rules on marked characteristics. Since this improvement provides a significant benefit for a small increase in running time, I recommend using it in compatible systems.

Finally, new methods should be explored to speed up the GATE processing and reduce the size of the processed passages. Pre-processing the entire corpus, prioritizing targets, and using the processed passages for other improvements should be the subject of future research. This year's question set should be thoroughly analyzed when TREC results are available.

7 Acknowledgements

This work was supported by an NSERC USRA

References

[1] T. Abou-Assaleh, N. Cercone, J. Doyle, V. Keselj, and C. Whidden, *DalTREC 2005 QA system Jellyfish: Mark-and-match approach to question answering.*, The Fourteenth Text REtrieval Conference (TREC 2005) Proceedings, 2005.

[2] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan, *GATE: A framework and graphical development environment for robust NLP tools and applications*, Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics, 2002.

[3] E. M. Voorhees, *Overview of TREC 2005*, The Fourteenth Text REtrieval Conference (TREC 2005) Proceedings, 2005.

[4] H.T. Dang E. M. Voorhees, *Overview of the TREC 2005 question answering track*, The Fourteenth Text REtrieval Conference (TREC 2005) Proceedings, 2005.

[5] Andrew Hickl, John Williams, Jeremy Bensley, Kirk Roberts, Ying Shi, , and Bryan Rink, *Question answering with LCC's Chaucer at TREC 2006*, The Fifteenth Text REtrieval Conference (TREC 2006) Proceedings (Gaithersburg, USA), 2006.

[6] B. Katz, G. Marton, S. Felshin, D. Loreto, B. Lu, F. Mora, . Uzuner, M. McGraw-Herdeg, N. Cheung, A. Radul, Y. Shen, and G. Zaccak, *Question answering experiments and resources*, The Fifteenth Text REtrieval Conference (TREC 2006) Proceedings (Gaithersburg, USA), 2006.

[7] Vlado Keselj, *DalTREC - Dalhousie TREC project*, Dalhousie University, `http://flame.cs.dal.ca/~trecacct/`, 2005.

[8] George A. Milleri, Richard Beckwith, Christiane Fellbaum, Derek Gross, K. Miller, and Randee Tengi, *Five papers on WordNet*, Available online at `ftp://ftp.cogsci.princeton.edu/pub/wordnet/5papers.pdf`, August 1993.

[9] openNLP, *openNLP tools*, `http://opennlp.sourceforge.net`, 2005.

[10] Lide Wu, Xuanjing Huang, Junkuo Cao, Xiafeng Yuan, and Yaqian Zhou, *FDUQA on TREC2006 QA track*, The Fifteenth Text REtrieval Conference (TREC 2006) Proceedings (Gaithersburg, USA), 2006.