Lymba's PowerAnswer 4 in TREC 2007

Dan Moldovan, Christine Clark, Mitchell Bowden
Lymba Corporation
Richardson, Texas 75080
moldovan@lymba.com

Abstract

This paper reports on Lymba Corporation's (a spinoff of Language Computer Corporation) participation in the TREC 2007 Question Answering track. An overview of the PowerAnswer 4 question answering system and a discussion of new features added to meet the challenges of this year's evaluation are detailed. Special attention was given to methods for incorporating blogs into the searchable collection, methods for improving answer precision, both statistical and knowledge driven, new mechanisms for recognizing named entities, events, and time expressions, and updated pattern-driven approaches to answer definition questions. Lymba's results in the evaluation are presented at the end of the paper.

1 Innovations for TREC 2007

New to TREC this year was a 175 GB collection of blog entries and an updated collection of 2.5 GB newswire articles. To meet the challenge of extracting answers from blogs PowerAnswer was prepared to search and process a collection of noisy data. For this reason mechanisms for filtering non-english text, filtering information-deficient articles and organizing blog entries into indexable documents were developed. To facilitate the required temporal indexing and processing of the AQUAINT 2 and Blog data, a new event and time detection system, called **Concept Tagger**, was introduced.

Perhaps the biggest change in PowerAnswer from the last few years consisted in the introduction of a new NER system, **Rose** that added new finer grained types for locations, quantities, and media.

Another innovation this year was the introduction of answer likelihood using question classes. By identifying question classes, improved language models can be built for these question types which in turn can be used during answer ranking.

Regarding list questions, COGEX logic prover was used for the first time as a re-ranker and candidate answer extractor during answer processing for list questions. New speciliazed types that consulted external resources were also integrated into the list question strategy. Both of these additions targeted the problem of increasing answer selection precision.

To maximize the coverage and robustness of the nugget extraction algorithms for "other" questions a hierarchy of nugget patterns and automatically derived generic answer patterns was developed.

What follows is a brief overview of PowerAnswer, followed by an indepth discussion of each of the innovations listed above and their impact on the results of the system. This paper will conclude with a summary of the overall results of PowerAnswer 4 at TREC 2007 and a discussion of future directions.

2 PowerAnswer 4 Overview

Lymba, formerly Language Computer Corporation, developed PowerAnswer as a fully-modular and distributed multi-strategy question answering system that integrates semantic relations, advanced inferencing abilities, syntactically constrained lexical chains, and temporal contexts. This section presents an outline of the system that was used in TREC 2007.

PowerAnswer contains a set of strategies that are selected based on advanced question processing, and each strategy is developed to solve a specific class of questions either independently or together. A Strategy Selection module automatically analyzes the question and chooses a set of strategies with the algorithms and tools that are tailored to the class of the given question.

Each strategy is a collection of components, (1) Question Processing (*QP*), (2) Passage Retrieval (*PR*), and (3) Answer Processing (*AP*). Each of these components con-

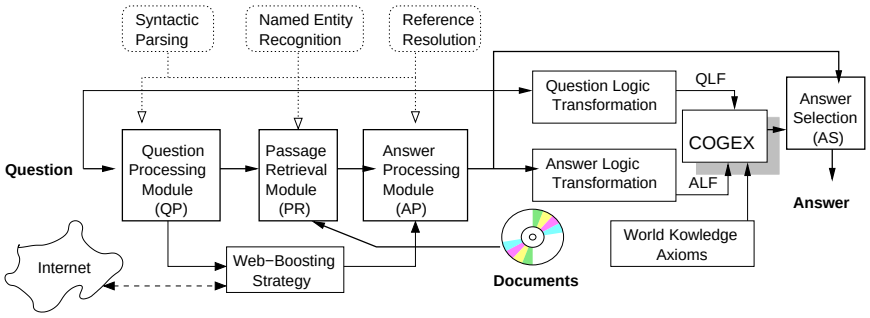

Figure 1: PowerAnswer 4 Architecture

stitute one or more modules, which interface to a library of generic NLP tools. These NLP tools are the building blocks of the PowerAnswer 4 system that, through a well-defined set of interfaces, allow for rapid integration and testing of new tools and third-party software such as IR systems, syntactic parsers, named entity recognizers, logic provers, semantic parsers, ontologies, word sense disambiguation modules, and more.

As illustrated in Figure 1, the role of the *QP* module is to (1) determine temporal constraints as defined by **Concept Tagger**, (2) detect the expected answer type, now extended with new types made available by **Rose**, (3) process any question semantics necessary such as roles and relations, (4) select the keywords used in retrieving relevant passages, (5) perform any preliminary questions as necessary for resolving question ambiguity, and (6) decide which **question class** to use when computing answer likelihood during answer ranking. The *PR* module ranks passages that are retrieved by the IR system, while the *AP* module extracts and scores the candidate answers based on a number of syntactic and semantic features such as keyword density, count, proximity, semantic ordering, roles, entity type, and, new to this year, an answer likelihood score, computed based on the language model associated with the class of the question. All modules have access to a syntactic parser, semantic parser, a named entity recognizer and a reference resolution system through Lymba's generic NLP tool libraries. To improve the answer selection, PowerAnswer takes advantage of redundancy in large corpora, specifically in this case, the Internet and Wikipedia. As the size of a document collection grows, a question answering system is more likely to pinpoint a candidate answer that closely resembles the surface structure of the question. These features have the role of correcting the errors in answer processing that are produced by the selection of keywords, by syntactic and semantic processing and by the absence of pragmatic information. The final decision for selecting answers is based on logical proofs from the inference engine COGEX (Moldovan, D. et al., 2006b).

3 Question Answering over Blog Data

New for TREC2007 was the dataset - updated newswire articles and the blog06 collection of weblog entries from the University of Glasgow. The blog collection presented its own unique set of challenges. Lymba's PowerAnswer had to overcome the size of the data, the organization of the blog entries, and the noisy nature of the text.

3.1 Collection Preparation

The raw data size of the collection is 175GB. Much of this is surrounding HTML (or XML) and many of the actual entries are quite small when separated. After a pipeline of processing steps, the final data size used by Lymba for TREC2007 was 13.1GB, or a reduction of *92.5%*. The final index used was 7GB.

To process the blog06 collection to be useful to PowerAnswer, we first parsed all the files to identify unique content, removing the duplicate entries. Secondly, we used an in-house language detection tool to identify and remove the non-English documents, spam documents and documents that had no retrievable text outside of the headline. This initial processing was done in a matter of 40 hours on 3 servers. Then we sent the remaining text through a limited set of Lymba's NLP toolset. The NLP tools used were distributed across 6 machines such that the processing of this subset of the data could be processed and indexed using Apache Lucene in 2 hours.

4 Temporal Event Processing

4.1 Overview of Concept Tagger

For 2007, we included the **Concept Tagger** module into our NLP processing pipeline. Concept Tagger detects the *events* present in a question or a candidate passage and labels them with their corresponding class, as seen in Table 1, and identifies a variety of temporal expressions, including absolute dates, times, durations and sets as well as resolving fuzzy temporal expressions, shown in Table 2. The **Concept Tagger** module uses a set of rules which operate on the full parse tree of the analyzed text.

Event Class	Question
Occurrence	
marry	*Who is he planning to marry?*
acquire	*What company acquired IMG in 2004?*
evacuation	*In the three months following the evacuation...*
State	
held	*In what city were ... held?*
Aspectual	
begin	*On what date did the court begin ...?*

Table 1: Examples of temporal classes detected by Concept Tagger.

Expression	Question
Abs. Date	
2004	*What company acquired IMG in 2004?*
Duration	
three months	*In the three months following...*
Sets	
each year	*How many grants...each year?*
Fuzzy	
now	*Where does Curveball now live?*

Table 2: Examples of temporal expressions identified by Concept Tagger.

All temporal expressions are normalized according to the TimeML TIMEX3 (TimeML Working Group 2005) standard making possible the extraction of accurate answers from passages which do not have words in common with the analyzed question. For example, Q249.5's correct answer was extracted from a passage which makes use of the frequency denoting concept *annually* whose normalized value (EVERY_P1Y) is identical to *each year*, concept present in the question.

Q249.5: *How many grants does the Fulbright Program award each year?*
P: The program named after the former Senator J. William Fulbright awards approximately **4,500** new grants annually.

The rich information extracted by **Concept Tagger** is used to detect event-event relations as well as time-event relations which are vital to the identification of relevant passages and extraction of accurate answers, as shown in Table 3. For this temporal relation identification, we adopted a hybrid approach which combines machine learning algorithms and manually developed set of rules which achieves an F-measure of 55% to 73% for the three subtasks of the TempEval Temporal Relation Identification Task (Task 15, SemEval 2007) (Min, C. et al., 2007). The feature set used to learn SVM models for temporal relations include information about the event (class, stem, part-of-speech, polarity, tense, aspect, modality, syntactic roles, semantic roles, cereference relations, etc.), the time (type, value, relation to the document creation time), the temporal signal, and the linguistic context (reporting, belief, volitional contexts). For TREC 2007, we enhanced this module to recognize biographical information (prevelant in news documents) by augmenting the set of syntactic patterns it uses to extract information. These new patterns were useful for passages such as the following which contain biographical temporal information:

Today [February 23, 2006] 's Birthdays: Bedrich Smetana, Bohemian composer (1824-1884); ... Kurt Weill, German-born American composer (1900-1950); ... (APW_ENG_20060223.0001)

The correct answer for question Q253.2 *In what year did Kurt Weill die?* was extracted from the passage shown above.

Relation	Question
Event-event	
during(be,write)	Q241.4: *How old was Jasper Fforde when he wrote his first Thursday Next novel?*
Time-event	
during(_date,begin)	Q227.2: *On what date did the court begin screening potential jurors?* Ans: Jury selection began six days ago [November 17, 2004] ... (APW_ENG_20041123.0053)

Table 3: Examples of event relations.

4.2 Performance Metrics

We evaluated the impact that **Concept Tagger** had on PowerAnswer's performance by analyzing the 93 factoid questions with temporal aspects distributed across all targets (with seven out of the sixteen event targets having explicit temporal information). All these questions require temporal processing with four of them necessitating temporal relation identification within the question which should be matched to the passage. PowerAnswer returned correct answers in *64.36%* of the remaining cases. The patterns for the biographical information were succesfully used for answering *3* temporal questions. The accurate resolution of fuzzy temporal expressions lead to the extraction of correct answers for *16* questions (two of them are shown in Table 4). The answer passages for the remaining *40* questions contained absolute temporal expressions which when linked to their corresponding events matched the answer type of the input questions (one example is listed in Table 4).

5 A New NER System: Rose

5.1 Overview of Rose

Lymba's named entity recognizer uses a three step process: (1) Pass the text through a pattern based grammar

259.2	Q: In what building was the 2005 World Snooker Championships held?
	P: Score on the 17th day of the World Snooker Championships at the **Crucible Theatre** here on Monday ... (AFP_ENG_20050502.0450)
259.3	Q: In what month did the 2005 Snooker World Championship start?
	P: ... the World Snooker Championship on Wednesday [**April** 27, 2005]. ... (APW_ENG_20050428.0349)
267.2	Q: In what year was the FISA court established?
	P: ... the FISA court Congress itself established in **1978** ... (BLOG06-20060210-015-0013857668)

Table 4: Examples of temporal expression resolutions in answer selection.

system, with rules maximizing recall, (2) Pass the grammar annotated data through an ML system based on (Carreras, X. et al, 2003), and (3) In the spirit of (Mikheev, A. et al, 1998) perform partial matching on the text.

Rose starts by invoking a traditional pattern matching and lexicon based information extraction engine. Input rule files are compiled into a graph representation and a depth first search is performed to see if a certain token starts a pattern match. The output of this step is a BIO labeled document. The set of named entity types consists of an extended set from previous competitions. Among new to this competition are types that represent various quantities, and specific locations.

Having text annotated by the grammars, the data is next input to a two pass system based on (Carreras, X. et al, 2003). The process is separated into annotating the existance of named entities using the BIO labeling scheme and classifying the recognized named entities. The first task is, essentially, chunk annotation. Instead of using AdaBoost, per (Carreras, X. et al, 2003), Rose employs Taku Kudo's YamCha (Kudo, T. and Matsumoto, Y. 2003) for the BIO task. What follows are the features extracted in a word window of ± 2:

- The surface form of the token
- The Part–of–Speech tag
- Whether the word's first character is capitalized
- 3 character prefixes and suffixes
- The output of the grammar system in BIO format, e.g. B-AWARD

A simple maximum entropy model is used to classify the identified named entities into PER, ORG, and LOC classes. The results are merged with the output of the grammar system. When the grammar system results in a subtype of the ML system, the more specific tag is preferred. When there is a conflict between the high level type returned by the ML system and the grammar system, the ML system output takes precedence.

Finally, we index all the named entities found in a document, create probable partial matches in the vein of (Mikheev, A. et al, 1998), and determine whether the partial match is correct based on the output of a maximum entropy model.

	Precision	Recall
Grammars (devel)	76.28%	62.56%
Grammars (test)	71.40%	58.29%
+ML NER (devel)	85.05%	87.98%
+ML NER (test)	73.20%	78.90%
+Partial (devel)	85.08%	89.55%
+Parital (test)	73.48%	81.91%

Table 5: Results from the named entity recognition system, Rose, on the MUC development and test sets (Restricted to ORG/LOC/PER types).

Type	Times used	Factoid Accuracy	List $F_{\beta=1}$
_media	20	80%	65.5%
_quantity_duration	13	76.9%	N/A
_facility_building	7	100%	48.85%
_quantity_age	6	50%	N/A
_quantity_length	5	80%	N/A
All	66	77.5%	56.92%

Table 6: Number of times each new type was used as an answer type. Types with less then 5 occurrences have been truncated.

5.2 Performance Metrics

The grammars were developed while looking over a large set of examples texts that included AQUAINT corpus data, WSJ, along with the data sets for MUC and ACE. Rose's lexicons were compiled from scratch from Wikipedia and other online resources, while the machine-learned components were trained on the MUC training data.

TREC answers of human entities usually require ranks and titles to be part of the returned answer. MUC annotations consider just the name itself to be the named entity. Our system returns "Secretary Ron Brown" instead of the desired "Ron Brown". "President Clinton" versus "Clinton". The results in Table 5 do not correct for this difference.

The impact of new types on the results was generally positive, as seen in Table 6. *77.5%* of the factoid questions where one of the new types was used was answered correctly. The average F mesaure for list questions with one of the new types were generally positive. Incorporating more specific question answering types helped the performance of PowerAnswer on these questions.

6 Answer Likelihood for Factoid Answer Selection

Answer ranking is a persistant challenge for any question answering system. PowerAnswer's primary semantic mechanism for extracting exact answers has been knowledge driven and enabled by COGEX (Moldovan, D. et al., 2006b). Due to the time complexity of parsing, currently there is an upper limit of between 10-20 candidate answer passages that are evaluated by COGEX. This restriction makes it very important that the answer processing and ranking modules up to that point are able to push the most correct answers into at least the top 20. For this reason Lymba experimented with clustering and language modelling techniques as part of the answer processing system similar to those reported by (Ko, J et al., 2007).

6.1 Generating Question Class Models

The goal of the answer likelihood component was to group questions from previous TRECs into classes, and then build language models for each class based on features extracted from the questions in the class as well as the answers judged as correct for each of these. The automatic formation of the question classes proved challenging. Lymba experimented with three methods for building the classes: (1) Generate regular expression style paraphrases for the questions in the training set (Clifton, T. et al., 2004), and then group them together based on their paraphrase identifiers, (2) Use hierarchical clustering (Manning, C. and Schutze, H. 1999) to organize the questions based on their expected answer type, most relevent keywords (as determined by the keyword selection algorithm), and named entity types, and (3) Group the questions together by their answer type.

6.2 Building the Question Class Models

Regardless of the method used for determining the question classes, the same approach was used to build language models for the classes. For both the questions in the classes and for each of the correctly judged answers as supplied by NIST, the following features were extracted:

1. Stemmed keywords from the question and the answer
2. Morphological alternations for each of the keywords
3. Named entity tags from both the question and the answers

From these features, a language model using good-turing smoothing was built with the SRILM (Stolcke, A. 2002) toolkit for each of the question classes. These models were then consulted during answer processing in order to compute the likelihood that an answer is in fact the correct response to a question.

6.3 Integration in the Answer Ranking Pipeline

The answer likelihood component was injected into PowerAnswer at two different stages: (1) During question processing a question was classified into one of the available question classes. In the case of the paraphrasing models, the paraphrase for the input question was generated, which was used to decide which model to consult later in the pipeline. (2) During answer processing, each of the semantic candidates retrieved after semantic matching are given a score to represent the probability or likelihood of the candidate sentence being in the language model for the question class chosen during question processing.

The log of the score of the answer likelihood was then added as a feature to the existing estimated relevance function embedded in PowerAnswer answer procesing (Moldovan, D. et al., 2004). From this point the top N candidates are passed to COGEX to re-rank the candidates based on how well the question is entailed by the given candidate answer.

6.4 Experimental Observations

For each of the three question class formation methods described above, Lymba ran a set of experiments on previous TREC test sets in order to determine which set of question classes performed the best. Leading into TREC 2007 it was empirically determined that the models based on grouping questions by answer types was most effective, and so was the configuration used for the TREC 2007 test set.

Our observations for this outcome include that for the models derived from the regular expression style paraphrases for the questions, the classes were too sparse as the software developed for this task was not able to generalize the patterns enough. The result was a large number of question classes with very few instances in them. For the models built from trying to cluster the classes using the named entities, n-gram features, the answer type, and the most relevant keywords of the questions, the clusters generated were noisy, and contained groups of questions that were not necessarily semantically similar. We hypothesize that the weighting schemes for the models needed more fine tuning. By simply grouping the questions by answer type, there is at least a guarantee that the expected focus of the questions in the class are the same, and so the classes generated tended to be more robust.

6.5 Performance Metrics

Although for the test set evaluation of the TREC 2007 questions, Lymba used the answer type driven models at run time, during post-TREC analysis, we benchmarked the system using two of the question class methods, eliminating the paraphrase based models since they were not able to generalize well enough for use at the time of this

writing. In the Table 7 below is a summary of the results over the TREC 2007 factoid questions. The row *Clusters* refers to the classes derived using hierarchical clustering. The row *Answer Type* refers to the classes derived by grouping questions by their answer types.

Model Type	% better	% worse	% stable
Clusters	19.3%	14.8%	65.9%
Answer Type	17.8%	11.8%	70.4%

Table 7: Results for Using Answer Likelihood in Answer Ranking

7 Improving Precision for List Questions

Lymba PowerAnswer's list question answering strategy is to attempt to maximize recall by returning as many answer candidates as possible during passage retrieval using a set of lexico-semantic alternations and relaxing the query to include the target keywords and the most relevent keywords from the primary question text. One major challenge the system faces once this result set is returned is to effectively filter the candidates such that all false positives are removed from the set. In the past the answer selection component of this strategy relied heavily on statistical selection algorithms. This method proved to be overly greedy and resulted in suboptimal precision. In an attempt to overcome the precision issue, Lymba deployed two new extensions to the list answer ranking components. The first was to capitalize on the wide coverage of lists found in Wikipedia, and to consult this as an authoritative source. The second was to tightly integrate COGEX, Lymba's logic prover for textual inference, into the answer selection process for questions in highly confusable answer classes, specifically those seeking lists of humans.

What follows is a description of each of these methods along with a summary of their performance impact on the overall TREC 2007 list question answering results.

7.1 Improving Precision Authoritative Resources

In previous years PowerAnswer successfully employed external resources for specialized answer types such as movies, songs, and books. Bots for searching *amazon.com* and *imdb.com* were executed whenever a question asking about a topic in the domain of books, songs, or movies was detected. The role of the bots was to identify candidate answers from these resources, and generate a dynamic lexicon to be consulted during the answer selection and ranking. Due to the increasing popularity and perceived authoritativeness of Wikipedia, Lymba chose to develop a similar bot framework for all questions whose expected answer type fell into the category of "_media", a new native type introduced by Lymba's NER system, Rose. Leading up to TREC our team explored using the lists in Wikipedia for questions seeking humans and countries, but decided to only deploy the media feature during the actual evaluation.

Google's "I'm Feeling Lucky" search was employed to locate relevent Wikipedia articles. During question processing, if the expected answer type assigned to the question is "_media", a bot named "searchWiki" is initialized with a query that includes the target keywords.In the case of the example for question **Q282.5: What are titles of Pamuk's works?**, the bot command is: *searchWiki.pl "Orhan Pamuk" _media data/lists/wiki*.

A results set containing the downloaded pages from Wikipedia that discuss *Orhan Pamuk*, are sent to a page parser that extracts double-quoted entities on lines that begin with * or — (unordered lists or tables) and recurses into pages that are linked with *main—..* and *see also—..* on the main page. The section names need to match one of *Filmography, Films, Discography, Albums, Songs, Bibliography, Books, Novels, or Works*

Finally, a list of entities pulled from these pages and sections are placed in an in-memory dynamic lexicon that is used to filter and boost list answer candidates during answer selection. All candidates that at least 90% match any of the entries in the dynamic lexicon are considered sound and are added to the final list of results for the input question. In this case the resulting answers were all entities extracted from Wikipedia and included:

1. The White Castle
2. Snow
3. My Name is Red
4. The New Life
5. Sessiz Ev
6. The Black Book

7.2 Exploiting Textual Entailment for Human Seeking List Questions

One of the weakness of PowerAnswer's list question answering strategy involved questions seeking lists of humans. This can be explained by the fact that in any given candidate passages there are many references to humans. Due to the greedy nature of the algorithms used to extract answer candidates from passages for list questions, the system was prone to select false positives as answers to human seeking questions. Because, in previous TRECs, it has been demonstrated that COGEX is an effective reranking tool (Harabagiu, S. et al, 2003; Moldovan, D. et al., 2004; Harabagiu, S. et al,, 2005; Moldovan, D. et al., 2006a) for factoid seeking questions, Lymba integrated COGEX, the logic prover directly into the list answer extraction filtering process in order to increase the precision for human seeking questions.

7.2.1 COGEX for LIST Questions

For any candidate passage returned by the passage retrieval engine that scores above a pre-defined threshold, all the human entities in that passage are returned as potential candidate answers for the question. Assertions are formed from the question such that each candidate is hypothesized to be the answer to the question. For questions of the form *Name X* or *List Y*, the reformulation into an assertion followed the pattern *candidateAnswer is one of X/Y*. For example, for question-answer pair: **Q248.3:** *Name officers of CSPI*, **A:** *Michael Jacobson*, the resuting assertion is: *Michael Jacobson is one of the officers of CSPI*.

Once each candidate answer is inserted in the question, COGEX checks if the newly derived assertions are entailed by the corresponding candidate answer passages. Only candidates that received an entailment score above the optimal threshold learned from past TREC evaluations are returned as valid answers to the list question. Consider the list question **Q254.5** *What women have worn Chanel clothing to award ceremonies?* (House of Chanel), a candidate answer passage:

(P_i) **Reese Witherspoon***'s gown was not vintage ... Golden Globe winner wearing the same glittery Chanel cocktail dress that* **Kirsten Dunst** *had worn to the awards in 2003.... seen asking Chanel President* **Maureen Chiquet** *at an after-party ...*

It contains three mentions of women, where only two of the assertions, *Kirsten Dunst* and *Maureen Chiquet* are returned as candidate answers. [1] In Table 8 the proof sketch for the assertions:

(1) *Kirsten Dunst has worn Chanel clothing to award ceremonies* (H_i^1) and

(2)*Maureen Chiquet has worn Chanel clothing to award ceremonies* (H_i^2)

is shown, along with their scores, thus demonstrating the process that lead to *Kirsten Dunst*'s selection as an answer and the elimination of *Maureen Chiquet*. Axiom A_1 (also shown in Table 8) is used by the prover in both cases, but for the H_i^2 assertion, the synactic and semantic dependencies between *Maureen Chiquet* and the verb *wear* cannot be inferred from the passage and, therefore, the prover relaxes one of the verb's arguments and drops the *agent* semantic relation. These score penalties cause the second assertion's score to drop below the threshold and *Maureen Chiquet* to be removed as an answer for the question **Q254.5**.

[1] Because *Reese Witherspoon*/*Witherspoon* was already selected as an answer based on a different supporting passage, it is disregarded during the processing of this passage and its corresponding candidate answers.

7.3 Performance Metrics

In the 2006 TREC evaluation, PowerAnswer's average precision was: *0.498* and this year jumped to *0.539*. Ten out of the eighty-five questions in the list test set were classified as media and so utilized the Wikipedia as a resource for answer selection/confirmation. The average precision for these questions was *0.655*, and so clearly contributed to the increase in precision this year. Additionally the approach, described in Section 7.2.1, proved to be effective as our f-score for humans increased from *0.372* to *0.458* and more importantly our average precision rose from *0.412* to *0.493*.

8 "Other" or Definition Questions

The inherent challenge of "other" questions in the TREC QA Track is the filtering and selection of interesting and novel nuggets from a large corpus. The passage recall for information about a target is typically overwhelming, and pruning these passages to pick the best nuggets is time consuming and difficult.

The traditional method employed by PowerAnswer to extract nuggets is to execute a definition pattern matching module. Previously, a list of over 200 positive and negative pre-computed patterns was loaded into memory. The target was inserted into these patterns and the resulting query submitted to an index including stopwords and punctuation. These are high-precision patterns that indicate information of a definitional nature.

To extend this method for TREC 2007, we developed a hierarchy of nugget patterns and automatically derived generic answer patterns. Questions are classified into this hierarchy and all answer patterns for the individual node and all parent nodes in the hierarchy of nuggets are selected and executed to construct a set of minimally-required information about the target depending on the class into which that target falls.

8.1 Deriving Answer Patterns from a Question Class Hierarchy

The nugget hierarchy is developed based on question classes discovered from previous TREC question sets, resulting in 35 target classes (e.g. animal, actor, musician, literature). These nodes are then arranged using the WordNet hierarchy with pre-selected upper nodes such as person, organization, event and other_entity. Each class has associated with it a set of minimal information that is customized to that class. For example, a person pattern set has information about full name, birth, death, place of birth, residence, occupation, etc. An event set has information about begin time, end time, duration, location, participants, etc. An organization holds information

P_i	...& wear_VB(e5,x15,x19) & _occurrence_NE(e5) & same_JJ(x25,x19) & glittery_JJ(x16,x19) & Chanel_NN(x17) & _human_NE(x17) & cocktail_dress_NN(x18) & nn_NNC(x19,x17,x18) & Kirsten_NN(x20) & Dunst_NN(x21) & nn_NNC(x22,x20,x21) & _human_NE(x22) & wear_VB(e6,x22,x19) & _occurrence_EV(e6) & to_TO(e6,x23) & award_NN(x23) & _occurrence_EV(x23) & in_IN(x23,x24) & Time_CTMP(BeginFn(x24),2003,1,1,0,0,0) & Time_CTMP(EndFn(x24),2003,12,31,23,59,59) ...& ask_VB(e8,x31,x44) & _i_action_EV(e8) & president_NN(x41) & maureen_NN(x42) & Chiquet_NN(x43) & nn_NNC(x44,x41,x42,x43) & _human_NE(x44) & at_IN(e8,x45) & after_party_NN(x45) & ...& THEME_SR(x19,e6) & AGENT_SR(x22,e6) & DURING_SR(e6,x23) & DURING_SR(x23,x24)...
H_i^1	Kirsten_NN(x1) & Dunst_NN(x2) & nn_NNC(x3,x1,x2) & _human_NE(x3) & wear_VB(e1,x3,x6) & _occurrence_EV(e1) & Chanel_NN(x4) & _human_NE(x4) & clothing_NN(x5) & nn_NNC(x6,x4,x5) & to_TO(e1,x9) & award_NN(x7) & ceremony_NN(x8) & nn_NNC(x9,x7,x8) & _occurrence_EV(x9) & DURING_SR(e1,x9) & THEME_SR(x6,e1) & AGENT_SR(x3,e1)
H_i^2	Maureen_NN(x1) & Chiquet_NN(x2) & nn_NNC(x3,x1,x2) & _human_NE(x3) & wear_VB(e1,x3,x6) & _occurrence_EV(e1) & Chanel_NN(x4) & _human_NE(x4) & clothing_NN(x5) & nn_NNC(x6,x4,x5) & to_TO(e1,x9) & award_NN(x7) & ceremony_NN(x8) & nn_NNC(x9,x7,x8) & _occurrence_EV(x9) & DURING_SR(e1,x9) & THEME_SR(x6,e1) & AGENT_SR(x3,e1)
A_1	cocktail_dress_NN(x1) -> clothing_NN(x1)

Table 8: COGEX's proofs

on number of members, for-profit or not, income, goals. Specializations such as musician contain patterns pertaining to instruments played, hit songs, and so on.

Answer nugget patterns are automatically generated using question-and-answer pairs from prior TREC question sets as seeds. Similar questions are grouped together and the textual patterns that form the answers are generalized into a number of patterns. Parts-of-speech can be generalized to form patterns with greater recall. Named entities are also specified in the patterns.

8.2 Examples of Use

Examples of automatically-generated and generalized person pattern, demonstrating some of the permutations that were discovered in the corpus:

- _var _BE _DT _nationality _JJ _profession
- _nationality _profession _var
- _nationality _JJ _profession _var
- _var (_nationality _profession

Where _var is a generalization for any human target. Sentences containing these patterns are selected from the index and added to a collection of candidate answer nuggets. The nuggets are filtered for similarity of information within the set and filtered to remove information contained in previous FACTOID or LIST answers. The sentences are parsed and then trimmed by the system such that the lowest tree node that contains the pattern remains, plus or minus some padding to reach a minimum word ratio.

Generic patterns at the top of the hierarchy are also useful in retrieving interesting nuggets. Examples from the entity node are:

- _var, which
- _var (_DT

The higher the pattern is in the Question Class Hierarchy, the lower the weighting of the final score. This is done to limit spurious matches as the more specific patterns will have higher precision than these more generic ones.

8.3 Performance Metrics

The new technique implemented for this year's TREC yielded a 67.26% increase in pyramid score over the OTHER questions in TREC 2006 (0.281 versus 0.168).

9 Results

Table 9 illustrates the final results of Lymba Corporation's results in the TREC 2007 QA track. The second column summarizes the accuracy, f-score, and pyramid scores for factoid, list, and others respectively.

	PowerAnswer 4
Factoid	0.706
List	0.479
Other	0.281
Overall	0.484

Table 9: Lymba's TREC 2007 Results

10 Conclusion and Future Directions

TREC 2007 highlighted the requirement that any useful question answering system must be able to accurately extract answers from very large collections consisting of mixed formats and even noisy data. PowerAnswer 4 was able to meet this challenge with its robust document processing and indexing tools, as well as with new algorithms that focused on precision. Future directions for PowerAnswer will include continued refinement of the answer likelihood work as well as a continued emphasis on the knowledge drive approaches inherent in the application of COGEX and the consultation of external resources for answer verification.

11 Acknowledgements

We would like to thank the researchers and engineers from Lymba Corporation for their invaluable efforts towards this work. This work, and also our broader research in QA, is supported in part by the Disruptive Technology Office's (DTO) Advanced Question Answering for Intelligence (AQUAINT) Program.

References

Carreras, X. ; Màrquez, L. ; Padró, L. 2003 A simple named entity extractor using AdaBoost In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL*

Clifton, T. ; Teahan, W. 2004 Bangor at TREC 2004: question answering track In *Proceedings of the thirteenth text retrieval conference*

Harabagiu, S. ; Moldovan, D. ; Clark, C. ; Bowden, M. ; Williams, J. ; Bensley, J. 2003 Answer Mining by Combining Extraction Techniques with Abductive Reasoning

Harabagiu, S. ; Moldovan, D. ; Clark, C. ; Bowden, M. ; Hickl, A. ; Wang, P. 2005 Employing Two Question Answering Systems in TREC-2005 In *Proceedings of the fourteenth text retrieval conference*

Ko, J. ; Si; L. ; Nyberg E. 2007 A Probabilistic Framework for Answer Selection in Question Answering In *Proceedings of NAACL HLT 2007*

Kudo, T. ; Matsumoto, Y. 2003 Fast methods for kernel-based text analysis In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, 24-31

Manning, C. ; Schutze, H 1999 Foundations of Statistical Natural Language Processing *The MIT Press*

Mikheev, A. ; Grover, C. ; Moens, M. 1998 Description of the LTG System Used for MUC-7 In *Proceedings of the 7th Message Understanding Conference*

Min, C.; Srikanth, M. ; Fowler, A. 2007 LCC-TE: A Hybrid Approach to Temporal Relation Identification in News Text IN *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*

Moldovan, D.; Bowden, M. ; Tatu, M. 2006 A Temporally-Enhanced PowerAnswer in TREC 2006 In *Proceedings of the fifteenth text retrieval conference*

Moldovan, D.; Clark, C. ; Harabagiu, S. ; Hodges, D. 2007 Cogex: A semantically and contextually enriched logic prover for question answering In *Journal of Applied Logic* Volume 5, Issue 1, March 2007, Pages 49-69

Moldovan, D.; Harabagiu, S.;Clark, C.;Bowden, M. 2004 PowerAnswer 2: Experiments and Analysis over TREC 2004 In *Proceedings of the thirteenth text retrieval conference*

Stolcke, A. 2002 SRILM – An Extensible Language Modeling Toolkit In *Proc. Intl. Conf. on Spoken Language Processing* vol. 2, pp. 901-904

TimeML Working Group 2005 The TimeML Working Group. 2005. The TimeML 1.2 Specification.