Strategies for Advanced Question Answering

Sanda Harabagiu and Finley Lacatusu
Language Computer Corporation
1701 N. Collins Ste. 2000
Richardson, TX 75080
{sanda, finley}@languagecomputer.com

Abstract

Progress in Question Answering can be achieved by (1) combining multiple strategies that optimally resolve different question classes of various degrees of complexity; (2) enhancing the precision of question interpretation and answer extraction; and (3) question decomposition and answer fusion. In this paper we also present the impact of modeling the user background on Q/A and discuss the pragmatics pf processing negation in Q/A.

1 Introduction

Our fundamental premise is that progress in Q/A cannot be achieved only by enhancing the processing components, but it also requires generating the best strategies for processing each individual question. Thus we believe that Q/A systems capable of successfully processing complex questions should employ multiple strategies instead of the current pipeline approach, consisting of (1) question processing, (2) passage retrieval and (3) answer selection. The pipeline architecture was reported in (Prager et al., 2000; Moldovan et al., 2000; Hovy et al., 2001). Recently, a novel approach based on combinations of multiple independent agents implementing different answer finding strategies (multi-strategy) and multiple search spaces (multiple-source) was developed by the IBM QA group (Chu-Carroll et al., 2003). In (Echihabi and Marcu, 2003) another form of combining strategies for advanced QA is proposed: (1) a knowledge-based Q/A implementation based on syntactic/semantic processing is combined using a maximum-entropy framework with (2) a statistical noisy-channel algorithm for Q/A and (3) a pattern-based approach that learn from Web data. In this project we propose a different form of finding optimal strategies of advanced QA which is based on (a) Question Decomposition, (b) Answer Fusion and feedback from (c) Interactive Q&A and (d) User Background Recognition.

We argue that all this new architectures operate under the assumption that there is a concept-based or pattern-based method for identifying the correct answer for any question that will be processed. However, we believe that there are complex questions that need first to be decomposed into simple questions, for which concept-based or pattern-based resolving techniques either exists or may be developed. For instance, when asking Q_1: "*How have thefts impacted on the safety of Russia's nuclear navy, and has the theft problem been increased or decreased over time?*" we may have series of simpler questions that decompose the question focus. One such example of simple question is Q_1^a: "*What specific instances of theft do we know about?*" – which is a list-question similar to those evaluated in the recent TREC tracks (Harabagiu et al., 2003). Related, simpler question is Q_1^b: "*What sort of items have been stolen?*". Question Q_1^a asks about instantiations of the theft events, whereas question Q_1^b inquires about the objects of the events. The decompositions may follow other arguments of the event predicates, e.g. – the agents in Q_1^c: "*Who are the perpetrators of these thefts?*" as well as specializations of the events, e.g. "economical impact" specializing one of the possible impacts of the thefts in the question Q_1^d: "*Do thefts have an economical impact on the naval bases?*". Furthermore, the concepts from the complex question need to be clearly understood, and often definition questions will be considered as decompositions that enable the processing of complex questions. The definition may involve entities from the complex question, e.g. Q_1^e: "*What is meant by nuclear navy?*" or events from the complex question, e.g. Q_1^f: "*What does 'impact' mean?*"

There are several criteria that guide question decomposition, which also determine the answer resolution strategies. The criteria are:

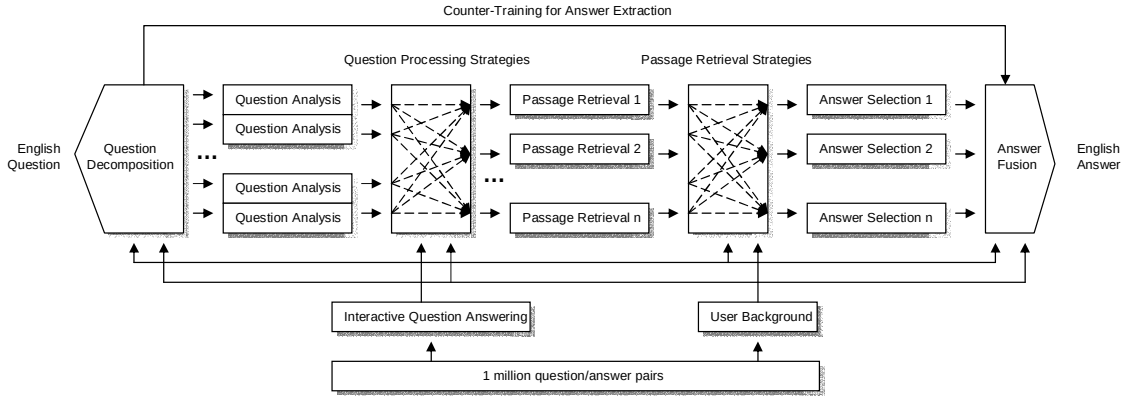

Figure 1: Answer resolution strategies

1. There are coordinations in the question format, suggesting decompositions along the constituents they coordinate. Coordinations may exist at: (a) question stem level, e.g. *"When and where did the thefts occur?"*; (b) at predicate level, e.g. *"How does one define an increase or a decrease in the theft problem?"*; (c) at argument level, e.g. *"To what degree do different thefts put nuclear or radioactive materials at risk?"*; (d) at question level, e.g. *"What specific instances of theft do we know about, and what are the sources of this information?*". Question decomposition by identifying coordinations involves: (a) disambiguation of conjunctives for identifying when they indicate separate questions as opposed to when they just coordinate constituents; (b) reference and ellipsis resolution of anaphoric expressions in the original question; (c) recognition of the relations between the resulting, decomposed questions, e.g. *contrast, reinforcement, mutual exclusion*.
2. The question asks about (a) a complex relation, e.g. *cause-effect, resultative, trend, likelihood*, (b) comparison with similar situations, or (c) elaboration of a state of affairs. Therefore the expected answer type is of complex nature and it requires definitions in the context of the complex scenario. The expected answer, recognized in a predicate from the question, determines the decomposition into (a) a definition question, (b) specializations of the predicate-concept, and (c) examples.
3. In order to search for the complex answer, elaborations of its arguments are needed. Such elaborations, called *argument-answer decompositions*, may involve (a) nested predicate-argument structures, (b) quantifications, or (c) instantiations.

When a complex question is processed, and is decomposed into a set of simpler questions which are analyzed independently. Each decomposed question may belong to a different class, for which certain strategies may be optimal. Such strategies implement the pragmatic processes that interact with the syntactic and semantic information that results from the derivation of: (1) expected answer types or structures, (2) name entities which are recognized, as well as (3) syntactic and semantic dependencies derived from the parsing of the question into predicate-argument structures. To be able to process the question precisely we are developing techniques that leverage a database of one million questions that have answers in a controlled corpus. This large database provides wide coverage of answer types and answer instances. It also enhances the retrieval, navigation and fusion of partial answers.

The challenge of creating a set of approximately one million question and answer pairs are twofold. First, the pairs need to be diverse in terms of difficulty, where difficulty can be defined in terms of answer type complexity (common, uncommon, requiring decomposition), answer granularity (concentrated within a small fragment or spread across several passages and documents), ease of matching (requiring both surface-text and deep semantic understanding). Second, the pairs should be reliable, i.e. each question must be associated with a correct answer. Our solution is a combination of collection and generation from semi-structured resources, followed by expansion and validation. We will generate the collection of QA pairs from Frequently Asked Questions (FAQ) files on various topics. We will develop a dedicated harvesting algorithm to identify FAQ's on the Web and extract the QA pairs.

The large database of questions also allows us to create a benchmark that will support the development of statistical techniques for Q/A. The architecture of the benchmark system is illustrated in Figure 1. Our system selects answers based on (1) question processing strategies; (2) passage retrieval strategies made possible by (3) question decomposition and (4) answer fusion. When a question is posed to the system, it is either decomposed on a set of simpler questions or it is processed in parallel with similar questions provided by the Interactive Question Answering component. Based on the user background, a set of similar questions may be selected and analyzed in parallel. Multiple strategies are available for retrieving relevant passages. The possible selections are once again dictated by feedback from

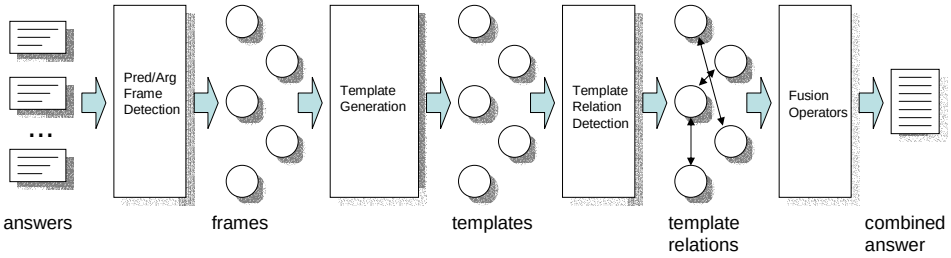

Figure 2: Answer fusion block architecture

interactions with the user. The relevant passages may also be combined on the basis of the same interactive and background information. We propose to study and develop several kernel methods that can operate in Support Vector Machines for determining the optimal strategies and compare the results with the Maximum Entropy combinations reported in (Echihabi and Marcu, 2003). The answer is produced by an answer fusion module that uses fusion operators. Since such operators are template-like, pattern acquisition methods may be employed for acquiring them.

The rest of the paper is organized as follows. The answer fusion strategies are presented in Section 2. Section 3details the methods for bootstrapping Question Answering. Section 4 describes the impact of the user background on the pragmatics of Q/A. Section 5 presents the problems engendered by processing negations in Question Answering. Section 6 summarizes the conclusions.

2 Answer Fusion, Ranking and Reliability

Given the size of today's very large document repositories, one can expect that any complex topic will be covered from multiple points of view. This feature is exploited by the question decomposition techniques, which generate a set of multiple questions in order to cover all of the possible interpretations of a complex topic. However, a set of decomposed questions may end up producing a disparate (and potentially contradictory) set of answers. In order for Q/A systems to use these collections of answers to their advantage, answer fusion must be performed in order to identify a single, unique, and coherent answer.

We view answer fusion as a three-step process. First, an open-domain, template-based answer formalization is constructed based on predicate-argument frames. Second, a probabilistic model is trained to detect relations between the extracted templates. Finally, a set of template merging operators are introduced to construct the merged answer. The block architecture for answer fusion is illustrated in Figure 2. The system functionality is demonstrated with the example illustrated in Figure 3.

Our method first converts the extracted answers into a series of open-domain templates, which are based on predicate-argument frames (Surdeanu et al, 2003). The next component detects generic inter-template relations. Typical "greedy" approaches in Information Extraction (Hobbs et al, 1997; Surdeanu and Harabagiu, 2002) use heuristics that favor proximity for template merging. The example in Figure 3 proves that this is not always the best decision, even for templates that share the same predicate and have compatible slots.

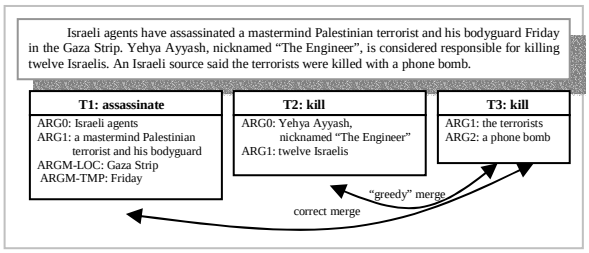

Figure 3: Examples of templates and template relations

2.1 Open-domain template representation

A key issue to the proposed approach is the open-domain template representation. While template-based representations have been proposed for information merging in the past (Radev and McKeown, 1998), they considered only domain-specific scenarios. Based on our recent successes with the extraction of predicate-argument frames (Surdeanu et al, 2003), we propose a template representation that is a direct mapping of predicate-argument frames. For example, the first template in Figure 3 is generated from the frame detected for the predicate "assassinate": the first slot – ARG0 – typically stands for subject or agent; the second slot – ARG1 – stands for the predicate object, and the modifier arguments ARGM-LOC and ARGM-TMP indicate the location and date of the event.

2.2 Detection of template relations

In this section, we introduce a probabilistic model for the detection of template relations that has been proven to infer better connectivity.

If the templates that are candidates for merging are selected entirely based on heuristics (Radev and McKeown, 1998; Surdeanu and Harabagiu, 2002), the application of fusion operators for QA is unreliable, due to their relatively weak semantic understanding of the

templates. The novelty in our approach is to precede template merging by the discovery of relations among templates.

We propose a novel matching approach based on template attributes that support relation detection for merging. The approach combines phrasal parsing, lemma normalization and semantic approximation (via WordNet lexical chains). For example, this approach detects that the attributes ARG1 of the first template ("assassinate") and ARG1 of the third template ("kill") from Figure 3 refer to the same entity, by matching "terrorist" with "terrorists". Moreover, the names of the templates ("assassinate" and "kill") are connected through a WordNet lexical chain.

A "greedy" detection procedure would incorrectly merge templates with the same name and a similar structure. The second and third templates from Figure 3, both named "kill", illustrate this case. Instead, we propose a novel probabilistic algorithm for relation detection. The algorithm computes a probability distribution of possible relations among entity templates, and retains those relations whose probabilities exceed a confidence threshold.

Operator	**Description**
CONTRADICTION	Two templates contain contradicting information, e.g. the same terrorist event is reported to have a different number of victims.
ADDITION	The second template introduces additional facts, e.g. one template indicates the location/date of a terrorist event while the second indicates number of victims.
REFINEMENT	The second template provides more refined information about the same event, e.g. the town instead of the country of location.
AGREEMENT	The templates contain redundant information. This operator is useful to heighten the answer strength.
GENERALIZATION	The two templates contain only incomplete facts that form an event only when combined.
TREND	The templates indicate similar patterns over time.
NO INFORMATION	The templates contain no useful information, e.g. unconfirmed event.

Table 1: Fusion Operators

2.3 Fusion Operators

The probabilistic model detects relations. A set of 7 template fusion operators is applied on the detected relations to generate the final set of templates. The operators are described in Table 1. The purpose of the fusion operators is to label the generic relations with the required merge operation, e.g. ADDITION, CONTRADICTION, TREND. For example, the templates T1 and T3 can be merged with the ADDITION operator. Optionally, the resulting template can be merged with template T2 with the weaker operator TREND, because they mark a similar type of event that takes place in the same location and date.

The generic template relations are labeled with one of the operators described in Table 1 with a machine learning approach based on Support Vector Machines (SVM) and a dedicated SVM kernel. SVMs are ideal for this task because they do not require an explicit set of features (a very complex endeavor in the planned open-domain environment), but localized kernels that provide a measure of template similarity. The labeled template relations direct the actual merging operation, which yields the final list of templates. Actual text can be generated from these templates, but this is beyond the goal of this paper.

3 Bootstrapping Question Answering

Two key components of modern QA systems are the identification of the answer type, and the extraction of candidate answers that are classified in the corresponding answer type category. For example, the question "*What weapons of mass destruction (WMD) does Iraq have?*" has the answer type "WMD" and accepts concepts such as "anthrax" as valid (but not necessarily correct) candidate answers. This approach provides an efficient implementation for the "exact answer" question answering spirit, but it is plagued by limited scalability.

We address the above scalability problem through several innovations: we develop a novel bootstrapping technique that increases significantly the coverage of the existing answer type categories. Furthermore, new answer type categories are created for concepts that can not be classified according to the currently available knowledge. In addition to the immediate application for answer extraction, the induced answer type knowledge is used to bootstrap the passage retrieval component, through intelligent query expansion.

Like most of the successful AQUAINT QA systems, LCC's system uses an answer type (AT) ontology for the classification of AT categories. The AT ontology is based on WordNet but can be extended with other open-domain or domain-specific categories. Instances of given categories are identified in answer passages using a modified version of the CiceroLite Named-Entity Recognizer (NER).

The first innovation in bootstrapping focuses on the capability of LCC's QA system to identify AT instances. The algorithm is summarized in Figure 4. The algorithm uses as input a very large set of question/answer pairs, and the existing AT ontology currently used by LCC's QA system. For each AT category, the algorithm adds the exact answers from the training question/answer pairs that share the same AT category to the BootstrappedLexicon, which is the lexicon generated as one outcome of this algorithm. Besides the lexicon, the algorithm induces a set of answer extraction patterns, BootstrappedPatterns, which guaran-

1. For each AT category:
   1.1. Extract all question/answer pairs from the training data set, where the question has this AT category.
   1.2. Add all exact answers to *BootstrappedLexicon*.
   1.3. *BootstrappedPatterns* = ∅
   1.4. Bootstrap loop:
      1.4.1. Generate all possible extraction patterns for *BootstrappedLexicon*.
      1.4.2. Score all extracted patterns.
      1.4.3. Find the best pattern P_{best} not in *BootstrappedPatterns.*
      1.4.4. Add P_{best} to *BootstrappedPatterns.*
      1.4.5. Add instances discovered with P_{best} to *BootstrappedLexicon*.

If termination condition not met go to 1.5.1.

Figure 4: Answer instance bootstrapping algorithm

tees the scalability of the proposed approach. BootstrappedPatterns is initialized to the empty set and is iteratively increased during the bootstrap loop. During the loop, the system scores all possible extraction patterns, and selects the best pattern to be added to BootstrappedPatterns. Concepts discovered with the newly extracted pattern are appended to BootstrappedLexicon, and the process repeats.

If a question/answer pair exists in the training set with "anthrax" as the exact answer, step 1.2 of the bootstrapping algorithm adds "anthrax" to BootstrappedLexicon. The bootstrap loop (step 1.4) mines the training documents for all possible patterns that contain anthrax. The best pattern selected is "deploy anthrax", which is generalized to "deploy ANY-WMD". This pattern is then used to extract other candidates for the WMD category, such as "smallpox", "botulinum" etc.

The algorithm illustrated in Figure 4 addresses the discovery of new instances for existing AT categories. A direct extension of this algorithm handles the situation when the discovered entities and patterns do not belong to a known category. The detection of new AT categories will be performed based on the AT word, which is the question concept that drives the selection of the AT. For example, the AT concept in the question: "What viral agent was used in Iraq?" is "viral agent", which does not exist in the current WordNet ontology. If the answer type concept does not exist in WordNet, the bootstrapping algorithm will create a distinct category for this concept. If the answer type concept exists in WordNet, the algorithm attaches the bootstrapped entities and patterns to the concept hypernym that provides the largest coverage without overlapping any other known categories. This approach is robust enough to function without word sense disambiguation: the algorithm explores all relevant synsets and selects the one that maximizes the above condition. For example, the answer type concept "fighter aircraft" from the question: "What fighter aircrafts are in use in the Iraqi army?" is mapped to the hypernym synset airplane (Sense #1), instead of vehicle (Sense #1), which overlaps with other vehicle categories such as cars.

3.1 Enhancing retrieval, navigation, and fusion

Answer accuracy is conditioned by the ability of the QA system to generate effective queries for the retrieval subsystem (Moldovan et al., 2003). Queries that are too restrictive will incorrectly narrow the search space, and fail to retrieve the relevant answer passages. An example of such a query is (biological AND agents AND Qaeda), which is generated for the question "What biological agents does al Qaeda possess?". This query will miss most of the relevant text passages since they do not include any explicit reference to biological agents.

The extensions to the AT ontology, described above, enable an intelligent query expansion based on two expansion resources: AT instances, and extraction patterns. More precisely, each question concept mapped under any of the AT categories is expanded with the instances and keywords from the extraction patterns associated with that category. In this case, the expanded query for the above question is: *((biological AND agents) OR (bacterial AND agent) OR (viral AND agent) OR (fungal AND agent) OR (toxic AND agent) OR botulism OR botulinum OR smallpox OR encephalitis OR (deploy)) AND (Qaeda).* This query illustrates two important requirements: the conversion of extraction patterns into keywords (e.g., "deploy" for "deploy ANY-WMD"); and the controlled expansion through selective keyword selection (e.g., for "biological agents").

3.2 Continuous updating of scenario knowledge

The bootstrapping algorithm described in the previous section is based on the large question/answer data set, which is largely open-domain. We consider a direct extension of this algorithm that automatically learns scenario knowledge by monitoring the user's browsing habits.

Question/answer pairs are extracted based on the user's feedback. These pairs form the seeds for a meta-bootstrapping loop, as illustrated in Figure 5. Similar documents – i.e. documents where the identical exact answer and the question keywords are identified – are produced from the relevant Q/A pairs. This process can

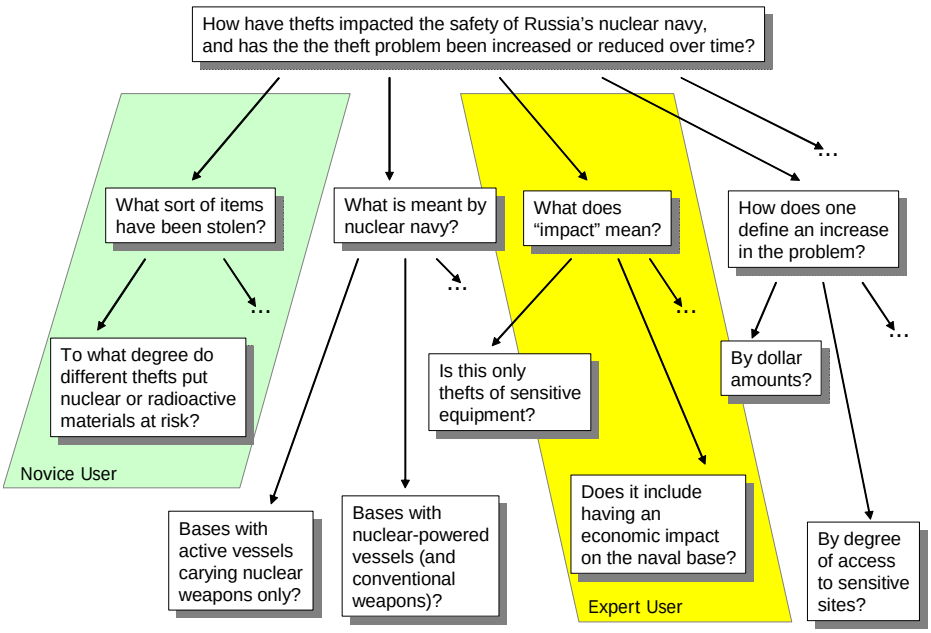

Figure 5: Example of different user selections from the generated question decomposition tree

be equally applied on the Web or on a static document collection. The bootstrapping algorithm described in Figure 5 is applied on the extracted documents. The inferred AT instances are further used to enrich the collection of considered documents, which forms the meta-bootstrapping loop.

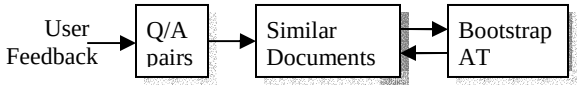

Figure 6:Scenario-specific meta-bootstrapping loop

4 User Background

Research in question answering, and in the more general field of information retrieval, has traditionally focused on building generic representations of the document content, largely independent of any subjective factors. It is important to note, however, that all users are different: not only do they have different backgrounds and expertise, but they also vary in their goals and reasons for using a Q/A system. This variety has made it difficult for systems to represent user intentions automatically or to make use of them in Q/A systems.

Figure 6 illustrates the inherent differences between system end-users. Since (by definition) a novice lacks the domain-specific knowledge available to an expert, we should expect a novice user to choose a path completely different than an expert user, leading to extremely different results for the same top level question.

4.1 Assessing User Background

We evaluate users via a discrete evaluation scale, which ranks users as novice, casual, or expert users based on how much background knowledge they have on the given topic. The approach classifies users based on the path chosen in the generated question decomposition tree.

This kind of characterization of user expertise can be used to reduce the exploration space generated through question decomposition. The most significant drawback of question decomposition is the exponential increase in the number of questions to be answered, which, to our knowledge, is not addressed by current QA research. We filter the generated question decomposition tree using the detected user expertise: for example, if the user is known to be an "expert", only the paths generated through "expert" decomposition - i.e. generated using significant world and topic knowledge – will be followed.

To be able to use the question decomposition tree for user classification, we must first classify the decomposition tree itself, i.e. the branches must be marked with one of the three discrete classification values. By shifting the classification problem from the (yet) abstract user background to the decomposition tree, we argue that the problem is simplified because we know how much background and world knowledge was necessary for the question decomposition. For example, to generate the "expert user" path in Figure 6, the system must have access to world knowledge that indicates that an "impact" can be economic, social etcetera and that

nuclear materials are sensitive equipment. We will quantify the amount of knowledge used for decomposition and label the generated branches accordingly.

Once a labeled decomposition tree is available, the user's background can be classified based on the selected path. The relevant answers (where "relevancy" can be explicitly requested from the user, or implicitly detected based on the documents visited) are mapped back to the corresponding questions, which provides a dynamic trace in the question decomposition tree. Using the tree structure and the classification labels previously assigned, we will train machine learning algorithms that will infer the final user expertise classification.

4.2 Representing User Background

We propose a new multi-modal approach for the representation of the user profile that includes concepts and relations in addition to terms.

Traditionally, the user profile (or background) has been represented as a term vector, derived from the previously relevant document (be it online or offline information). Under this approach, each profile P, is represented as: $P = ((t_1, w_1), (t_2, w_2), \ldots, (t_n, w_n))$, where t_i are terms from relevant documents and w_i are term weights, typically computed with the tf * idf metric.

Our approach is novel in two regards. The first innovation stems from the observation that it is common for one user to explore multiple topics even during the same session. For example, an analyst interested in the current Iraq situation, must explore topics related to military action, peace keeping, and terrorism. Hence the one vector representation for the profile P is clearly insufficient. In the proposed representation, the profile P is represented as a set of vectors $p_1, p_2, \ldots, p_n$, where: $p_i = ((t_{i1}, w_{i1}), (t_{i2}, w_{i2}), \ldots(t_{im}, w_{im}))$, i = 1, 2, …, n, and m is the size of vector p_i. We expect the number and size of the profile vectors to change dynamically. When a new document is marked as relevant, the document vector is either: (a) merged with an existing profile, if their similarities are higher than a given threshold, or (b) used to generate a new profile. Profile vectors are removed based on negative feedback: if a document vector similar to an existing profile receives continuous negative feedback the corresponding profile is deleted in order to keep the profile synchronized with the user's current interest patterns. We believe this profile representation to be flexible enough to accommodate all expertise levels, from novice to expert. For example, the expert user's background will consist of multiple vectors; each specializes on a clear, domain-specific direction, while the novice user's profile will most likely contain fewer vectors with more generic terms.

The second innovation includes concepts and relations in addition to lexical terms in the user profile. A preliminary analysis of the CNS documents indicates that "al" is among the most frequent terms, but, by itself, "al" is considered a stop word by most information retrieval systems. However, the significance of the term becomes evident when the complete concept, "al Qaeda" is considered. This observation indicates that semantic information is significant for the representation of the user profile. In addition to indexing entities, we index generic of entity-to-entity relations that are significant, and often the goal, of the intelligence analyst's work.

5 Processing Negation in Question Answering

Although all human systems of communication represent negation in some format, the issue of how best to address negation in open-domain Q/A still remains an open research question. Previous Q/A systems have dealt with negation by filtering the retrieved answer and eliminating answers that share key terms with the query but are irrelevant for the reasons of negation (Martinovic, 2002; Attardi et al, 2001) or by constructing relational databases to query the answers can handle negation in the question since the scope is clearly defined in the relational database (Jung and Lee, 2001). However, neither of these systems has dealt with the central problem that negation poses for Q/A: determining the scope of the negation context. Consider the following examples:

a. Which countries did not vote for the Iraq war resolution in the Security Council?

b. Which countries did not provide help to the coalition during the Gulf War in 1991?

c. What planets have no moon?

In question (a), the scope of negation only includes the countries that were members of the Security Council during the Iraq war resolution that were able to vote but did not. However, examples (b) and (c) are ambiguous with respect to the scope of negation. In question (b), the scope could encompass the whole world, or all the countries in the Middle East that should have provided help but did not. In question (c), even more entities can be included under the scope of negation: all of the planets in the solar system or even all of the planets in the entire universe (including planets that are not yet discovered).

In order for a Q/A system to answer questions like (b) or (c), the scope of negation must first be determined. We initially propose to develop empirical studies for recognizing the most frequent cases of negation: e.g. the "no" negation – "with no terrorists, the world would be safer", the "nothing" negation – "the inspectors found nothing", and other core cases of local negation – e.g. "thefts did not occur at the beginning". We

shall complement our methods of recognizing negation in textual sentences by analyzing various syntactic and semantic contexts of negation, e.g. adverbial negation – "the president never leaves the White House without the Secret Service approval".

In addition, we assume that when a speaker is formulating a question to find out whether a proposition is true or false, s/he formulates the question with the form of the proposition which would be the most informative if it turned out to be true. We expect that if a question has the form of negation, the speaker believes that the negative answer is the most informative. Using such hypotheses, we argue that in a negation question, if the scope is ambiguous, like in (b) or (c), then we can solve the ambiguity by choosing the scope that will be more informative for the user.

Given these assumptions, we propose that negation can be addressed in Q/A in three ways:

By using the user background. In questions like (b) above, if the user background is terrorism, then we can limit the scope of the countries to those who have been linked to terrorism.

By interacting with the user. If no user background can be established, as in question (c), we expect to use dialogue techniques to enable the user can specify the relevant scope.

By finding cues from the answers to the positive question. Finally, we expect to be able to use a combination of heuristics and information extraction methods to deduce the answer to a negative question from the answers to the corresponding positive question. For example, when searching for the answer to the positive analog of question (c), we can limit the scope of the negation to the solar systems where there are planets with moons.

6 Conclusions

In this position paper we discuss several strategies that enhance the pragmatic process involved in the interpretation and resolution of complex questions. In support of our claims we present a Q/A architecture used to benchmark the impact of (1) question decompositions following several criteria; (2) answer fusion which composes a unique, coherent answer from the partial answers extracted for each decomposed question; (3) modeling of user background and (4) processing of negation in questions and/or answers. Additionally, we present a bootstrapping algorithm that enhances the precision of factual Q/A. We argue that each of these enhancements would allow us to advance the state-of-the-art in Q/A and will enable us to correctly process complex questions.

References

A. L. Berger, S. A. Della Pietra and V. J. Della Pietra. *A maximum entropy approach to natural language processing.* Computational Linguistics, 22(1):39-72, March 1996.

Jennifer Chu-Carroll and Sandra Carberry. *Generating Information-Sharing Subdialogues in Expert-User Consultation*. In Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95), pages 1243-1250, Montreal, Canada, 1995.

Michael Collins. *A New Statistical Parser Based on Bigram Lexical Dependencies*. In Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics, ACL-96, pages 184-191, 1996.

Abdessamad Echihabi and Daniel Marcu. *A Noisy-Channel Approach to Question Answering*. In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-2003), Sapporo, Japan, 2003.

Christiane Fellbaum (Editor). *WordNet – An Electronic Lexical Database*. MIT Press, 1998.

C.J. Fillmore and C.F. Baker and H. Sato. *The FrameNet Database and Software Tools*. In M. González-Rodríguez and C.P. Suárez-Araujo, Eds. In Proceedings of the Third International Conference on Language Resources and Evaluation. Las Palmas, Spain, 2002.

Michael Fleischman, Eduard Hovy and Abdessamad Echihabi. *Offline Strategies for Online Question Answering: Answering Questions Before They Are Asked*. In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-2003), pages 1-7, Sapporo, Japan, 2003.

Sanda Harabagiu, Marius Paşca and Steven Maiorano. *Experiments with Open-Domain Textual Question Answering*. In Proceedings of the 18th International Conference on Computational Linguistics (COLING-2000), pages 292-298, Saarbrucken, Germany, 2000.

Sanda Harabagiu, Dan Moldovan, Christine Clark, Mitchell Bowden, John Williams and Jeremy Bensley. *Answer Mining by Combining Extraction Techniques with Abductive Reasoning*. In Proceedings of the 12th Text Retrieval Conference (TREC 2003).

Jerry R. Hobbs, Doug E. Appelt, John Bear, David Israel, Megumi Kameyama, Mark Stickel and Mabry Tyson. *FASTUS: A Cascaded Finite-State Transducer for Extracting Information Natural-Language Text*. In Finite State Language Processing, Edited by

Emmanuel Roche and Yves Schabes, MIT Press, 1997.

E.H. Hovy, U. Hermjakob, C.-Y. Lin, "The Use of External Knowledge in Factoid QA", TREC-10 conference, November 2001.

Dan I. Moldovan, Sanda M. Harabagiu, Marius Paşca, Rada Mihalcea, Roxana Gîrju, Richard Goodrum and Vasile Rus. *The Structure and Performance of an Open-Domain Question Answering System*. In Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000), 2000.

John Prager, Eric Brown, Anni Coden and Dragomir Radev. *Question-answering by predictive annotation*. In Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, pages: 184-191, Athens, Greece, 2000.

Dragomir R. Radev and Kathleen McKeown. *Generating natural languages summaries from multiple online sources*. Computational Linguistics, 24(3):469-500, 1998.

M. Surdeanu and S. Harabagiu, "Infrastructure for Open-Domain Information Extraction", in *Proceedings of the Conference for Human Language Technology (HLT-2002)*, pages 325-330, March 2002.

Mihai Surdeanu, Sanda Harabagiu, John Williams and Paul Aarseth, "Using Predicate-Argument Structures for Information Extraction", In Proceedings of ACL 2003.