Answer Mining from On-Line Documents

Marius Paşca and **Sanda M. Harabagiu**
Department of Computer Science and Engineering
Southern Methodist University
Dallas, TX 75275-0122
{mars,sanda}@engr.smu.edu

Abstract

Mining the answer of a natural language open-domain question in a large collection of on-line documents is made possible by the recognition of the *expected answer type* in relevant text passages. If the technology of retrieving texts where the answer might be found is well developed, few studies have been devoted to the recognition of the answer type.

This paper presents a unified model of answer types for open-domain Question/Answering that enables the discovery of exact answers. The evaluation of the model, performed on real-world questions, considers both the correctness and the coverage of the answer types as well as their contribution to answer precision.

1 Introduction

Answer mining, a.k.a. textual Question/Answering (Q/A), represents the task of discovering the answer to an open-domain natural language question in large text collections. Answer mining became a topic of significant recent interest, partly due to the popularity of Internet Q/A services like AskJeeves and partly due to the recent evaluations of domain-independent Q/A systems organized in the context of the Text REtrieval Conference (TREC)[1]. The TREC evaluations of fully automatic Q/A systems specified two restrictions: (1) there is at least one document in the test collection that contains the answer to a test question; and (2) the answer length is either 50 contiguous bytes (short answers) or 250 contiguous bytes (long answers). These two requirements intentionally simplify the answer mining task, since the identification of the *exact* answer is left to the user. However, given that the expected information is recognized by inspecting text snippets of relatively small size, the TREC Q/A task took a step closer to *information retrieval* rather than *document retrieval*. Moreover, the techniques developed to extract text snippets where the answers might lie paved the way to a unified model for answer mining.

To find the answer to a question several steps must be taken, as reported in (Abney et al., 2000) (Moldovan et al., 2000) (Srihari and Li, 2000):

- First, the question semantics needs to be captured. This translates into identifying (i) the *expected answer type* and (ii) the *question keywords* that can be used to retrieve text passages where the answer may be found.

- Secondly, the index of the document collection must be used to identify the text passages of interest. The retrieval method either employs special operators or simply modifies boolean or vector retrieval. Since the expected answer type is known at the time

[1]The Text REtrieval Conference (TREC) is a series of workshops organized by the National Institute of Standards and Technology (NIST), designed to advance the state-of-the-art in information retrieval (IR)

of the retrieval, the quality of the text passages is greatly improved by filtering out those passages where concepts of the same category as the answer type are not present.

- Thirdly, answer extraction takes place by combining several features that take into account the expected answer type.

Since the expected answer type is the only information used in all the phases of textual Q/A, its recognition and usage is central to the performance of answer mining.

For an open-domain Q/A system, establishing the possible answer types is a challenging problem. Currently, most of the systems recognize the answer type by associating the *question stem* (e.g. *What*, *Who*, *Why* or *How*) and one of the concepts from the question to a predefined general category, such as PERSON, ORGANIZATION, LOCATION, TIME, DATE, MONEY or NUMBER. Since many of these categories are represented in texts as named entities, their recognition as possible answers is enabled by state-of-the-art Named Entity (NE) recognizers, devised to work with high precision in Information Extraction (IE) tasks. To allow for NE-supported answer mining, a large number of semantic categories corresponding to various names must be considered, e.g. names of cars, names of diseases, names of dishes, names of boats, etc. Furthermore, a significant number of entities are not unique, therefore do not bear names, but are still potential answers to an open-domain question. Additionally, questions do not focus only on entities and their attributes; they also ask about events and their related entities.

In this paper we introduce a model of answer types that accounts for answers to questions of various complexity. The model enables several different formats of the *exact answer* to open-domain questions and considers also the situation when the answer is produced from a number of different document sources. We define formally the answer types to open-domain questions and extend the recognition of answer types beyond the question processing phase, thus enabling several feed-back mechanisms derived from the processing of documents and answers.

The main contribution of the paper is in providing a unified model of answer mining from large collections of on-line documents that accounts for the processing of open-domain natural language questions of varied complexity. The hope is that a coherent model of the textual answer discovery could help developing better text mining methods, capable of acquiring and rapidly prototyping knowledge from the vast amount of on-line texts. Additionally, such a model enables the development of intelligent conversational agents that operate on open-domain tasks.

We first present a background of Q/A systems and then define several classes of question complexity. In Section 3 we present the formal answer type model whereas in Section 4 we show how to recognize the answer type of open-domain questions and use it to mine the answer. Section 5 presents the evaluation of the model and summarizes the conclusions.

2 Background

Open-Domain Question/Answering

To search in a large collection of on-line documents for the answer to a natural language question we need to know (1) *what* we are looking for, i.e. the expected answer type; and (2) *where* the answer might be located in the collection. Furthermore, knowing the answer type and recognizing a text passage where the answer might be found is not sufficient for extracting the exact answer. We also need to know the dependencies between the answer type and the other concepts from the question or the answer. For example, if the answer type of the TREC question

QT: How many dogs pull a sled in the Iditarod?

is known to be a *number*, we also need to be aware that this number must quantify the dogs harnessed to a sled in the Iditarod games and not the number of participants in the games.

Capturing question or answer dependencies can be cast as a straightforward process of mapping syntactic trees to sets of binary head-modifier relationships, as first noted in (Collins, 1996). Given a parse tree, the head-child of each syntactic constituent can be identified based on a simple set of rules used to train syntactic parsers, cf. (Collins, 1996). Dependency relations are established between each leaf corresponding to the head child and the leaves of its constituent sib-

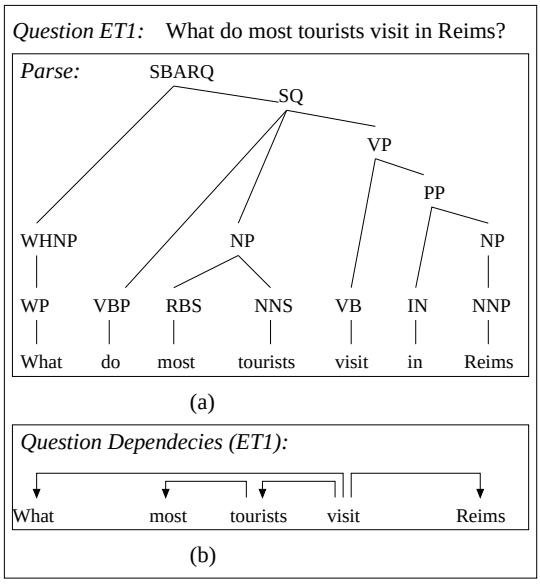
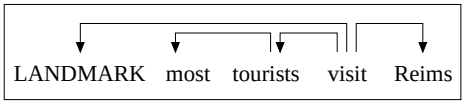


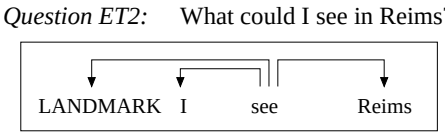
Figure 1: Example of TREC test question

lings that are not stop words, as illustrated by the mapping of Figure 1(a) into Figure 1(b). Unlike in IR systems, question stems are considered content words. When question dependencies are known (Harabagiu et al., 2000) proposed a technique of identifying the answer type based on the semantic category of the question stem and eventually of its most connected dependent concept. For example, in the case of question *ET1*, illustrated in Figure 1, the answer type is determined by the ambiguous question stem *what* and the verb *visit*. The answer type is the object of the verb *visit*, which is a place of attraction or entertainment, defined by the semantic category LANDMARK. The answer type replaces the question stem, generating the following dependency graph, that can be later unified with the answer dependency graph:

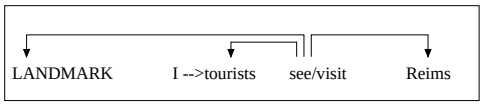


However syntactic dependencies vary across question *reformulations* or *equivalent* answers made possible by the productive nature of natural language. For example, the dependency structure of *ET2*, a reformulation of question *ET1* differs from the dependency structure of *ET1*:

Due to the fact that verbs *see* and *visit* are synonyms (cf. WordNet (Miller, 1995)) and pronoun *I* can be read a possible visitor, the dependency



structures of *ET1* and *ET2* can be mapped one into another. The mapping is produced by unifying the two structures when lexical and semantic alternations are allowed. Possible lexical alternations are synonyms or morphological alternations. Semantic alternations consist of hypernyms, entailments or paraphrases. The unifying mapping of *ET1* and *ET2* shows that the two questions are equivalent only when *I* refers to a visitor; other readings of *ET2* being possible when the referent is an investigator or a politician. In each of the other readings, the answer type of the question would be different. The unifying mapping of *ET1* and *ET2* is:



Similarly, a pair of equivalent answers is recognized when lexical and semantic alternations of the concepts are allowed. This observation is crucial for answer mining because:

1. it establishes the dependency relations as the basic processing level for Q/A; and

2. it defines the search space based on alternations of the question and answer concepts.

Consequently, lexical and semantic alternations are incorporated as feedback loops in the architecture of open-domain Q/A systems, as illustrated in Figure 2.

To locate answers, text passages are retrieved based on keywords assembled from the question dependency structure. At the time of the query, it is unknown which keywords can be unified with answer dependencies. However, the relevance of the query is determined by the number of resulting passages. If too many passages are generated, the query was too broad, thus is needs a specialization by adding a new keyword. If too few passages were retrieved the query was too specific, thus one keyword needs to be dropped. The relevance feedback based on the number of retrieved passages ends when no more keywords can be added or dropped. After this, the unifications of the question and answer dependencies

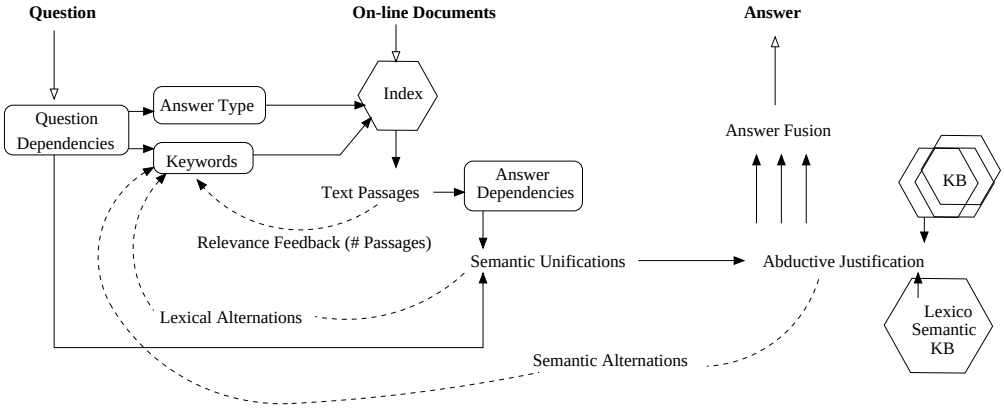


Figure 2: A diagram of the feedbacks supporting Open-Domain Q/A

is produced and the lexical alternations imposed by unifications are added to the list of keywords, making possible the retrieval of new, unseen text passages, as illustrated in Figure 2.

The unification of dependency structures allows erroneous answers when the resulting mapping is a sparse graph. To justify the correctness of the answer an abductive proof backchaining from the answer to the question must be produced. Such abductive mechanisms are detailed in (Harabagiu et al., 2000). Moreover, the proof relies on lexico-semantic knowledge available from WordNet as well as rapidly formated knowledge bases generated by mechanisms described in (Chaudri et al., 2000). The justification process brings forward semantic alternations that are added to the list of keywords, the feedback destination of all loops represented in Figure 2.

Mining the exact answer does not always end after extracting the answer type from a correct text snippet because often they result only in partial answers that need to be fused together. The fusion mechanisms are dictated by the answer type.

Question Complexity

Open-Domain natural language questions can also be of different complexity levels. Generally, the test questions used in the TREC evaluations were qualified as *fact-based* questions (cf. (Voorhees and Tice, 2000)) as they mainly were short inquiries about attributes or definitions of some entity or event. Table 1 lists a sample of TREC test questions.

The TREC test set did not include any question

Where is Romania *located*? **Europe**
Who *wrote* "Dubliners"? **James Joyce**
What is the *wingspan* of a condor? **9 feet**
What is the *population* of Japan? **120 million**
What *king* signed the Magna Carta? **King John**
Name a flying *mammal*. **bat**

Table 1: TREC test questions and their exact answers (boldfaced)

that can be modeled as Information Extraction (IE) task. Typically, IE templates model queries regarding *who* did an event of interest, *what* was produced by that event, *when* and *where* and eventually *why*. The event of interest is a complex event, like terrorism in Latin America, joint ventures or management successions. An example of template-modeled question is:

What management successions occurred at IBM in 1999?

In addition, questions may also ask about developments of events or trends that are usually answered by a text summary. Since data producing these summaries can be sourced in different documents, summary fusion techniques as proposed in (Radev and McKeown, 1998) can be employed. Template-based questions and summary-asking inquiries cover most of the classes of question complexity proposed in (Moldovan et al., 2000). Although the topic of natural language open-domain question complexity needs further study, we consider herein the following classes of questions:

- *Class 1:* Questions inquiring about entities, events, entity attributes (including number),

event themes, event manners, event conditions and event consequences.

- *Class 2:* Questions modeled by templates, including questions that focus only on one of the template slots (e.g. *"What managers were promoted last year at Microsoft?"*).

- *Class 3:* Questions asking for a summary that is produced by fusing template-based information from different sources (e.g. *"What happened after the Titanic sunk?"*).

Since (Radev and McKeown, 1998) describes the summary fusion mechanisms, Class 3 of questions can be reduced in this paper to Class 2, which deals with the processing of the template.

3 A Model of Answer Types

This section describes a knowledge-based model of open-domain natural language answer types (ATs). In particular we formally define the answer type through a quadruple
$AT = [$CATEGORY, DEPENDENCY, NUMBER, FORMAT$]$.
The CATEGORY is defined as one of the following possibilities:

1. one of the tops of a predefined ANSWER TAXONOMY or one of its nodes;
2. DEFINITION;
3. TEMPLATE; or
4. SUMMARY.

For expert Q/A systems, this list of categories can be extended. The DEPENDENCY is defined as the question dependency structure when the CATEGORY belongs to the ANSWER TAXONOMY or is a DEFINITION. Otherwise it is a template automatically generated. The NUMBER is a flag indicating whether the answer should contain a single datum or a list of elements. The FORMAT defines the text span of the exact answer. For example, if the CATEGORY is *DIMENSION*, the FORMAT is *<Number><Measuring Unit>*.

The ANSWER TAXONOMY was created in three steps:
Step 1 We devise a set of top categories modeled after the semantic domains encoded in the WordNet database, which contains 25 noun categories and 15 verb categories. The top of each WordNet hierarchy corresponding to every semantic category was manually inspected to select the most representative nodes and add them to the tops of he ANSWER TAXONOMY. Furthermore we have added open semantic categories corresponding to named entities. For example Table 2 lists the named entity categories we have considered in our experiments. Many of the tops of the ANSWER TAXONOMY are further categorized, as illustrated in Figure 3. In total, we have considered 33 concepts as tops of the taxonomy.

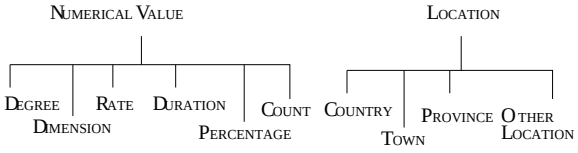


Figure 3: Two examples of top answer hierarchies.

Step 2 The additional categorization of the top ANSWER TAXONOMY generates a many-to-many mapping of the Named Entity categories in the tops of the ANSWER TAXONOMY. Figure 4 illustrates some of the mappings.

date	*time*	*organization*	*city*
product	*price*	*country*	*money*
human	*disease*	*phone number*	*continent*
percent	*province*	*other location*	*plant*
mammal	*alphabet*	*airport code*	*game*
bird	*reptile*	*university*	*dog breed*
number	*quantity*	*landmark*	*dish*

Table 2: Named Entity Categories.

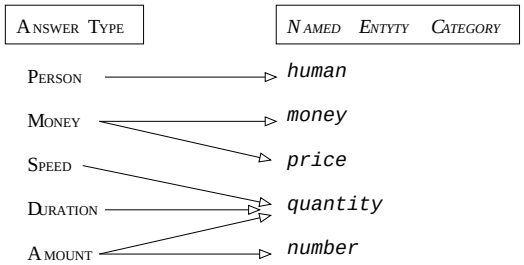


Figure 4: Mappings of answer types in named entity categories.

Step 3: Each leaf from the top of the ANSWER TAXONOMY is connected to one or several Word-

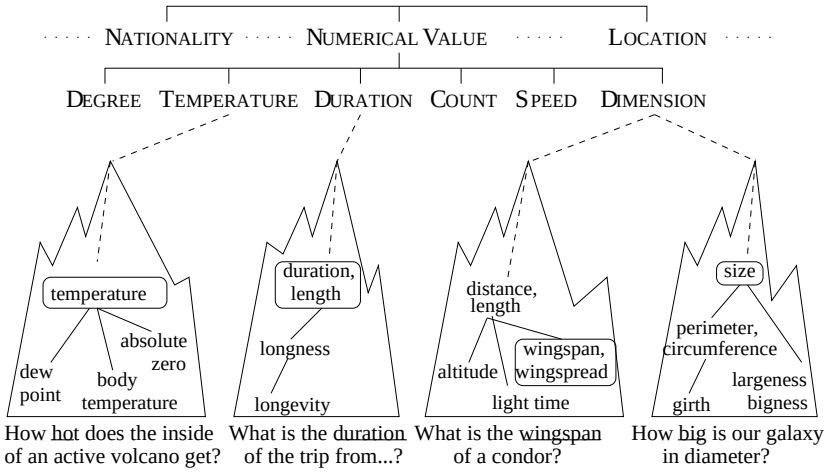


Figure 5: Fragment of the ANSWER TAXONOMY.

Net subherarchies. Figure 5 illustrates a fragment of the ANSWER TAXONOMY comprising several WordNet subhierarchies.

4 Answer Recognition and Extraction

In this section we show how, given a question and its dependency structure, we can recognize its answer type and consequently extract the exact answer. Here we describe four representative cases.
Case 1: The CATEGORY of the answer type is DEFINITION when the question can be matched by one of the following patterns:

(Q-P1):*What* {*is*|*are*} <*phrase_to_define*>?
(Q-P2):*What is the definition of* <*phrase_to_define*>?
(Q-P3):*Who* {*is*|*was*|*are*|*were*} <*person_name(s)*>?

The format of the DEFINITION answers is similarly dependent on a set of patterns, determined as the head of the <*Answer_phrase*>:

(A-P1):[<*phrase_to_define*> {*is*|*are*}] <*Answer_phrase*>
(A-P2):[<*phrase_to_define*>, {*a*|*the*|*an* <*Answer_phrase*>}]
(A-P3):[<*phrase_to_define*> –] <*Answer_phrase*>

Case 2: The dependency structure of the question indicates that a special instance of a concept is sought. The cues are given either by the presence of words *kind*, *type*, *name* or by the question stems *what* or *which* connected to the object of a verb. Table 3 lists a set of such questions and their corresponding answers. In this case the answer type is given by the subhierarchy defined by the node from the dependency structure whose adjunct is either *kind*, *type*, *name* or the question stem. In this situation the CATEGORY does not belong to the top of the ANSWER TAXONOMY, but it is rather dynamically created by the interpretation of the dependency graph.

For example, the dynamic CATEGORY *bridge*, generated for *Q204* from Table 3, contains 14 member instances, including *viaduct*, *rope bridge* and *suspension bridge*. Similarly, question *Q581* generates a dynamic CATEGORY *flower*, with 470 member instances, comprising *orchid*, *petunia* and *sunflower*. For dynamic categories all member instances are searched in the retrieved passages during answer extraction to detect candidate answers.

Case 3: In all other cases, the concept related to the question stem in the question dependency graph is searched through the ANSWER TAXONOMY, returning the answer type as the top of it hierarchy. Figure 5 illustrates several questions and their answer type CATEGORY.

Case 4: Whenever the semantic dependencies of several correct answers can be mapped one into another, we change the CATEGORY of the answer type into TEMPLATE. The slots of the actual template are determined by a three step procedure, that we illustrate with a walk-through example corresponding to the question *What management successions occurred at IBM in 1999?*:
Step 1: For each pair of extracted candidate

Q204: What type of *bridge* is the Golden Gate Bridge?
Answer: the Seto Ohashi Bridge, consisting of six *suspension bridges* in the style of Golden Gate Bridge.
Q267: What is the name for *clouds* that produce rain?
Answer: Acid rain in Cheju Island and the Taean peninsula is carried by *rain clouds* from China.
Q503: What kind of *sports* team is the Buffalo Sabres?
Answer: Alexander Mogilny hopes to continue his *hockey* career with the NHL's Buffalo Sabres.
Q581: What *flower* did Vincent Van Gogh paint?
Answer: In March 1987, van Gogh's "*Sunflowers*" sold for $39.9 million at Christie's in London

Table 3: TREC test questions and their answers. The exact answer is emphasized.

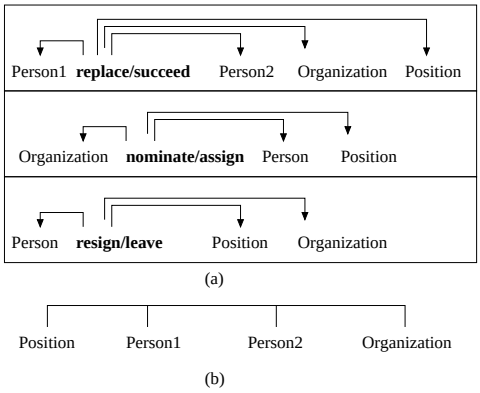


Figure 6: Dependencies that generate templates.

answers unify the dependency graphs and find common generalizations whenever possible. Figure 6(a) illustrates some of the mappings.

Step 2: Identify across mappings the common categories and the trigger-words that were used as keywords. In Figure 6(a) the trigger words are boldfaced.

Step 3: Collect all common categories in a template and use their names as slots. Figure 6(b) illustrates the resulting template.

This procedure is a reverse-engineering of the mechanisms used generally in Information Extraction (IE), where given a template, linguistic patterns are acquired to identify the text fragments having relevant information. In the case of answer mining, the relevant text passages are known. The dependency graphs help finding the linguistic rules and are generalized in a template.

To be able to generate the template we also need to have a way of extracting the text where the answer dependencies are detected. For this purpose we have designed a method that employs a simple machine learning mechanism: the perceptron. For each text passage retrieved by the keyword-based query we define the following seven features:

- rel_{SP} the number of question words matched in the same phrase as the answer type CATEGORY;
- rel_{SS} the number of question words matched in the same sentence as the answer type CATEGORY;
- rel_{FP}: a flag set to 1 if the answer type CATEGORY is followed by a punctuation sign, and set to 0 otherwise;
- rel_{OCTW}: the number of question words matches separated from the answer type CATEGORY by at most three words and one comma;
- rel_{SWS}: the number of question words occurring in the same order in the answer text as in the question;
- rel_{DTW}: the average distance from the answer type CATEGORY to any of the question word matches;
- rel_{NMW}: the number of question words matched in the answer text.

To train the perceptron we annotated the correct answers of 200 of the TREC test questions. Given a pair of answers, in which one of the answers is correct, we compute a relative comparison score using the formula:

$$\begin{aligned} rel_{pair} \quad = \quad & w_{SWS} \times \Delta rel_{SWS} + w_{FP} \times \Delta rel_{FP} \\ & + w_{OCTW} \times \Delta rel_{OCTW} + w_{SP} \times \Delta rel_{SP} \\ & + w_{SS} \times \Delta rel_{SS} + w_{NMW} \times \Delta rel_{NMW} \\ & + w_{DTW} \times \Delta rel_{DTW} + threshold \end{aligned}$$

The perceptron learns the seven weights as well as the value of the threshold used for future tests on the remaining 693 TREC questions. Whenever the relative score is larger than the threshold, a passage is extracted as a candidate answer. In our experiments, the performance of the perceptron surpassed the performance of decision trees for answer extraction.

5 Evaluations and Conclusion

To evaluate our answer type model we used 693 TREC test questions on which we did not train the perceptron. Table 4 lists the breakdown of the answer type CATEGORIES recognized by our model as well as the coverage and precision of the recognition. Currently our ANSWER TAXONOMY encodes 8707 concepts from 129 WordNet hierarchies, covering only 81% of the expected answer types. This shows that we have to continue encoding more top concepts in the taxonomy and link them to more WordNet concepts.

The recognition mechanism had better precision than coverage in our experiments. Moreover a relationship between the coverage of answer type recognition and the overall performance of answer mining, as illustrated in Table 4. Some of the test questions are listed in Tables 1 and 3. The experiments were conducted by using 736,794 on-line documents from *Los Angeles Times*, *Foreign Broadcast Information Service*, *Financial Times AP Newswire*, *Wall Street Journal* and *San Jose Mercury News*.

CATEGORY (# Questions)	*Precision*	*Coverage*
DEFINITION (64)	91%	84%
Top ANSWER TAXONOMY (439)	79%	74%
Dynamic answer category (17)	86%	79%
TEMPLATE (14)	93%	65%

# ANSWER Taxonomy Tops	Answer Type Coverage	Q/A Precision
8	44%	42%
22	56%	55%
33	83%	78%

Table 4: Evaluation results.

The experiments show that open-domain natural language questions of varied degrees of complexity can be answered consistently from vast amounts of on-line texts. One of the applications of a unified model of answer mining is the development of intelligent conversational agents (Harabagiu et al., 2001).

Acknowledgement

This research was supported in part by the Advanced Research and Development Activity (ARDA) grant 2001*H238400*000 and by the National Science Foundation CAREER grant CCR-9983600.

References

S. Abney, M. Collins, and A. Singhal. 2000. Answer extraction. In *Proceedings of ANLP-2000*, pages 296–301, Seattle, Washington.

V.K. Chaudri, M.E. Stickel, J.F. Thomere, and R.J. Waldinger. 2000. Reusing prior knowledge: Problems and solutions. In *Proceedings of AAAI-2000*, Austin, Texas.

M. Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the ACL-96*, pages 184–191, Copenhagen, Denmark.

S. Harabagiu, M. Paşca, and S. Maiorano. 2000. Experiments with open-domain textual question answering. In *Proceedings of COLING-2000*, Saarbrucken, Germany.

S. Harabagiu, M. Pasca, and F. Lacatusu. 2001. Dialogue management for interactive question answering. In *Proceedings of FLAIRS-2001*. To appear.

G. Miller. 1995. WordNet: a lexical database. *Communications of the ACM*, 38(11):39–41.

D. Moldovan, S. Harabagiu, M. Paşca, R. Mihalcea, R. Gîrju, R. Goodrum, and V. Rus. 2000. The structure and performance of an open-domain question answering system. In *Proceedings of ACL-2000*, Hong Kong.

D. Radev and K. McKeown. 1998. Generating natural language summaries from multiple on-line resources. *Computational Linguistics*, 24(3):469–500.

R. Srihari and W. Li. 2000. A question answering system supported by information extraction. In *Proceedings of ANLP-2000*, Seattle, Washington.

E.M. Voorhees and D.M. Tice. 2000. Building a question-answering test collection. In *Proceedings of the 23rd International Conference on Research and Development in Information Retrieval (SIGIR-2000)*, Athens, Greece.