High Performance Question/Answering

Marius A. Paşca
Department of Computer Science and Engineering
Southern Methodist University
Dallas, TX 75275-6221
mars@engr.smu.edu

Sanda M. Harabagiu
Department of Computer Sciences
University of Texas
Austin, TX 78712
sanda@cs.utexas.edu

ABSTRACT

In this paper we present the features of a Question/Answering (Q/A) system that had unparalleled performance in the TREC-9 evaluations. We explain the accuracy of our system through the unique characteristics of its architecture: (1) usage of a wide-coverage answer type taxonomy; (2) repeated passage retrieval; (3) lexico-semantic feedback loops; (4) extraction of the answers based on machine learning techniques; and (5) answer caching. Experimental results show the effects of each feature on the overall performance of the Q/A system and lead to general conclusions about Q/A from large text collections.

1. INTRODUCTION

Question/Answering (Q/A) is an IR paradigm for discovering answers to an open-domain natural language question from a large collection of texts. The task of Q/A did not appear in the vacuum. The first Q/A systems were developed in the late 70s as interfaces to problem-solving systems (e.g. STUDENT [14] solved algebra problems, LUNAR [15] allowed geologists to ask questions about moon rocks). The tradition of employing Q/A systems as interfaces to expert systems, using large knowledge bases and reasoning mechanisms continues even today. The systems developed within the DARPA High Performance Knowledge Bases Project (HPKB) were recently evaluated through Q/A exercises in a narrow domain [2]. Unfortunately such evaluations discard textual information that can be processed automatically in favor of knowledge engineered manually. With the advent of massive collections of on-line documents, the manual translation of textual information into knowledge bases covering large numbers of domains is impractical. Moreover, the size of these collections imposes new Q/A paradigms, robust and scalable, unlike the Natural Language Processing (NLP) systems used in HPKB (e.g. MIT's START [7]) that impose massive manual annotations of the texts.

In addition, several theories of Q/A have been developed in the context of NLP or cognitive sciences. First, we have the conceptual theory of Q/A, proposed by Wendy Lehnert in [8], with an associated question taxonomy; and then, we have the mechanisms for generating questions developed by Graesser & al. in [4]. However, these theories are not open-ended. They did not assume large-scale resources, and did not rely on high-performance parsers, named entity recognizers or information extractors, tools developed in the last decade under the impetus of the TIPSTER program. Perhaps the most influential body of research for the current open-domain textual Q/A paradigm comes from the Information Extraction (IE) technology. IE systems participating in the Message Understanding Conferences (MUCs) have been quite successful as extracting information from newswire messages and filling database templates with information pertaining to the events of interest. Typically, the templates model queries regarding *who* did *what* to *whom*, *when* and *where*, and eventually *why*. Like knowledge-based Q/A systems, IE systems depend on domain knowledge.

When the TREC-8 Q/A track initiated evaluations of factual questions whose answers can be found within 2 GBytes[1] text collections, a new Q/A paradigm was imposed. Five answers of either 50 contiguous bytes (*short* answer) or 250 bytes (*long* answer) were expected to be returned to each open-domain natural language question. Finding these answers is made possible by a special form of retrieval, in which the question keywords occur in the same document passage, as first reported in [5], [10]. In addition, to become answer candidates, these passages must contain at least one concept of the same semantic category as the *expected answer type*. This novel constraint imposes the processing the question semantics before starting the retrieval process. The systems that performed well in the two TREC Q/A evaluations considered very likely that the expected answer type is a named entity, e.g. representing a person, an organization a location or some other category. For this purpose, these systems employed named entity recognizers, first developed for the IE technology, which operate with human-like performance.

In this paper we argue that the recognition of the expected answer type is not the only factor that determines high accuracy of a Q/A system. In fact, we show that keyword alternations of lexical, morphological and semantic nature are equally important in retrieving the correct answer. These alternations are implemented as feedback for several retrieval loops, showing that in Q/A systems, the IR component has central usage. In addition we show that answer extraction based on machine learning techniques surpasses empirical methods. We also describe our answer caching techniques,

[1] In TREC-9, the size of the collection became 3GBytes.

SIGIR'01, September 9-12, New Orleans, Louisiana, USA..

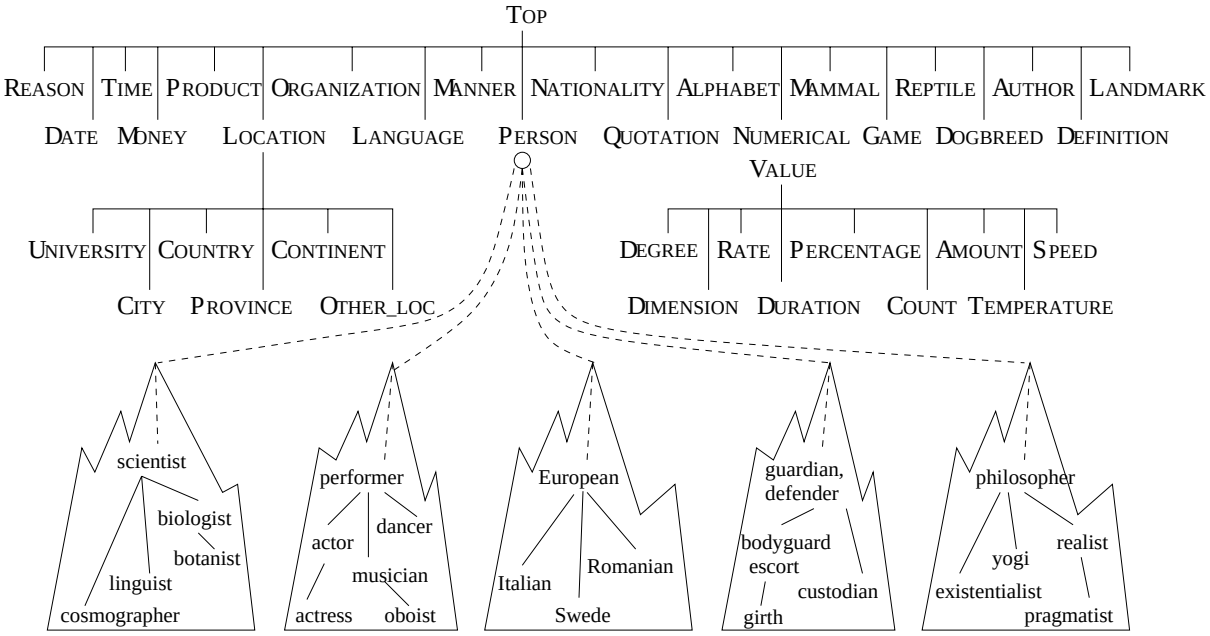


Figure 1: Answer Type Taxonomy

that take into account the possibility that a similar question was previously processed.

The rest of the paper is organized as follows.Section 2 presents the answer type hierarchy used for finding the expected answer type of a vast majority of the test questions. Section 3 elaborates on the retrieval feedbacks that help discover the candidate answer passages whereas Section 4 details the machine learning experiments for extracting answers. Section 5 evaluates the effects of the features of our Q/A system whereas Section 6 summarizes the conclusions.

2. THE ANSWER TYPE TAXONOMY

A text passage containing a candidate answer contains not only some of the question keywords, but necessarily one concept of the same semantic category as the concept inquired by the natural language question, be it a person's name, a number, a date, a measure, a location or an organization. We define the semantic category of the answer as the *expected answer type*. For open-domain questions inquiring only about entities or events or some of their attributes or roles, as was the case with the test questions in TREC-8 and TREC-9, an off-line taxonomy of answer types can be built by relying on vast lexico-semantic resources such as WordNet [9]. The WordNet 1.6 database encodes more than 100,000 English nouns, verbs, adjectives and adverbs organized in conceptual synonym sets, known as *synsets*. Moreover, the nouns and verbs are further organized in hierarchies by Is-A relations and classified into 25 noun categories and 15 verb categories.

In building the answer type taxonomy, we followed three steps:

∘ *Step 1:* For each semantic category of nouns or verbs we manually examined the most representative conceptual nodes and added them as tops of the ANSWER TYPE TAXONOMY. Moreover, we added open semantic categories corresponding to named entities. The tops of the ANSWER TYPE TAXONOMY are illustrated in Figure 1. It can be noted that some of the tops (e.g. LOCATION or NUMERICAL VALUE) are further categorized, mostly because they subsume distinct semantic types correspond to categories identified by our named entity recognizer.

∘ *Step 2:* Since often the expected answer type is a named entity, a many-to-many mapping between the named entity categories and the tops of the ANSWER TYPE TAXONOMY is required. Figure 2 illustrates some of the implemented mappings.

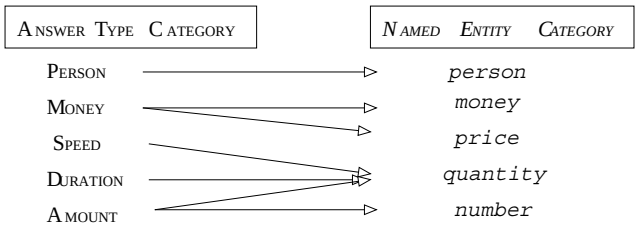


Figure 2: Many-to-many mappings between named entities and tops of the Answer Type Taxonomy

For example, either an AMOUNT, a DURATION or the SPEED are recognized as *Amount* expressions by a named entity recognizer, whereas concepts of type MONEY are identified either as *Money* or *Price* expressions by our named entity recognizer.

∘ *Step 3:* Each leaf of the tops implemented in the ANSWER TYPE TAXONOMY is manually linked to one or several subhierarchies from WordNet. Figure 1 illustrates only some of the hierarchies that generate an answer type classified as PERSON. Similar links are encoded for each leaf of the ANSWER TYPE TAXONOMY. These links connect abstract concepts, identified at *Step 1* with sub-hierarchies from WordNet, in which concepts are represented as synsets.

Currently, the ANSWER TYPE TAXONOMY encodes 8707 En-

glish concepts, both nouns, verbs and adjectives, that help recognize the expected answer type of an open-domain natural language question. The vast majority of the concepts are nouns, since most of the leaves of the ANSWER TYPE TAXONOMY are connected to WordNet noun sub-hierarchies. Moreover, a leaf from the ANSWER TYPE TAXONOMY may be connected simultaneously to (a) noun sub-hierarchies, (b) verb sub-hierarchies or (c) adjectival satellites encoded in the WordNet database. For example the PRODUCT leaf from the ANSWER TYPE TAXONOMY is connected to both nouns from the sub-hierarchy synset {*artifact, artefact*} and to verbs from the sub-hierarchy of {*manufacture, fabricate, construct*}.

Table 1: The most connected tops of the ANSWER TYPE TAXONOMY.

Concept	Number of connections
PERSON	36
ORGANIZATION	16
PRODUCT	10
DIMENSION	9
CITY	6

Adjectival synsets and their WordNet satellites are connected through theVALUE-OF lexico-semantic relations encoded in WordNet. For example, *far*, *small* and *tall* are connected to the DIMENSION leaf due to the fact that they are values of *distance*, *size* and *stature or height*, respectively. Similarly *rich* is linked to MONEY and *many* to COUNT.

The ANSWER TYPE TAXONOMY encodes 153 connections to WordNet sub-hierarchies. 130 of these connections link tops of the taxonomy to WordNet noun sub-hierarchies. Most of the tops are linked to one or two sub-hierarchies. Table 1 illustrates the tops of the ANSWER TYPE TAXONOMY that are connected to the most numerous WordNet sub-hierarchies.

The availability of such an ANSWER TYPE TAXONOMY solves only half of the problem. When an open-domain natural language question is asked, we need to identify the question word(s) that determine the *expected answer type*. Some of the question stems, when present, are unambiguous, e.g. *who* always asks for a person or an organization. However, most of the question stems are highly ambiguous, e.g. *what* can start a question asking about anything. Some of the early systems (e.g. [13] [1] [11]) implemented rules that established the correspondence between the question stem and the named entity category. To our knowledge, our Q/A system is the first to employ a wide coverage ANSWER TYPE TAXONOMY complemented by a robust mechanism of identifying the question word that determines the expected answer type.

To determine the expected answer type we parse the question and search for the word that has a binary head-modifier dependency to the question stem. For this purpose we employ our own implementation of Collins' parser [3] and use the dependency relations learned when training the probabilistic parser. For example, in the case of TREC-9 question *Q712: What do tourists visit in Reims?*, the parse tree of the question is shown in Figure 3.

Each non-terminal in the parse tree at level two and above represents a syntactic constituent. For each possible constituent, there are rules identifying the head child and propagating it to its parent (see Figure 4). In the case of question *Q712* the propagation identifies the stem *what* to be a dependent of the verb *visit*.

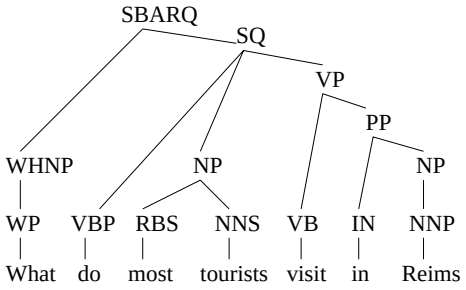


Figure 3: Parse tree of a TREC question.

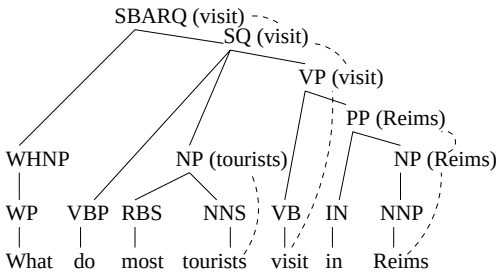


Figure 4: Propagation of labels in the question parse tree.

In this case, the expected answer type is a typical object of the verb *visit*, mapped in the top LANDMARK from the ANSWER TYPE TAXONOMY. Concepts categorized as landmarks are museums, palaces, castles, cathedrals, etc. The identification of the question word(s) determining the answer type based on syntactic parses is more accurate than empirical methods of associating semantic categories to the head of the first phrase or to trigger-words, as employed in IE systems. Our methodology was successful in more than 90% of the TREC test questions, failing only when the coverage of our taxonomy was not sufficient. Table 2 illustrates some TREC questions for which the expected answer type cannot be determined unless a parse of the question is performed and an extensive answer taxonomy is available. Table 2 also shows examples of questions for which only a phrasal parse is sufficient.

Table 2: Examples of TREC questions and their corresponding expected answer types.

Parse	Question	Expected Answer Type
Full	*Q003: What does the Peugeot company manufacture?*	PRODUCT
Full	*Q012: How much did Manchester United spend on players in 1993?*	MONEY
Phrasal	*Q265: What's the farthest planet from the sun?*	PLANET
Phrasal	*Q209: Who invented the paper clip?*	PERSON

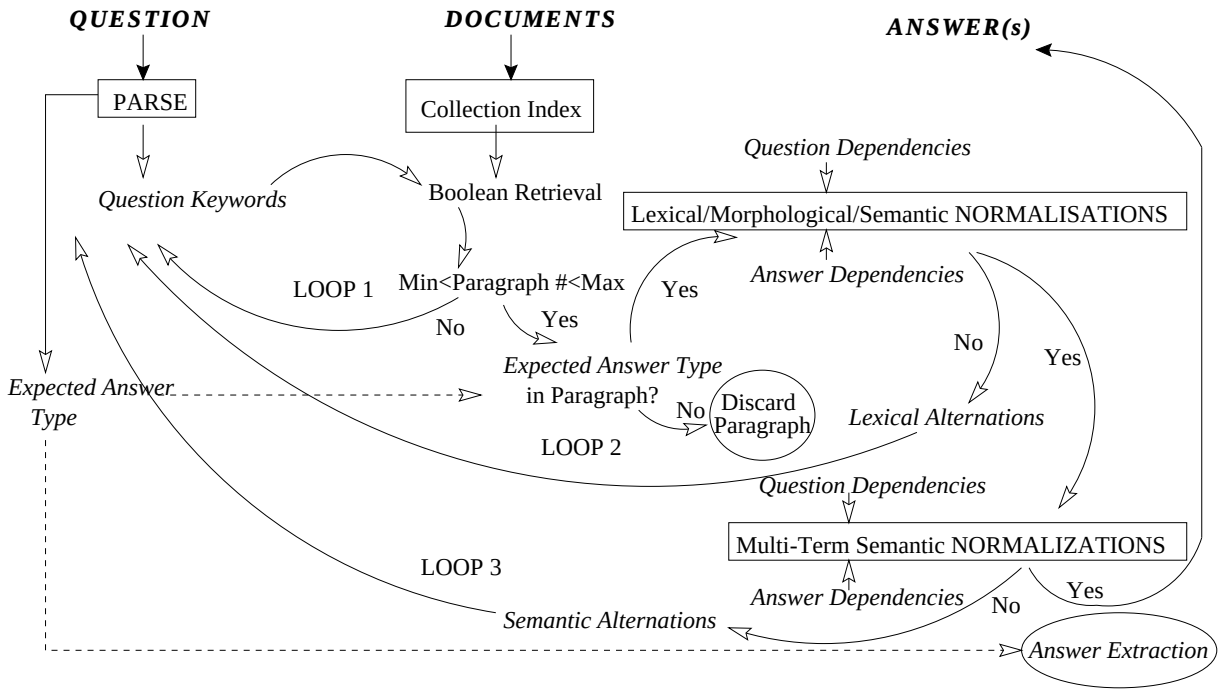


Figure 5: Retrieval Feedbacks in a Q/A System

In general the expected answer type is returned as the top of the ANSWER TYPE TAXONOMY. There are however a few exceptions, that were implemented in our system. One of them is represented by definition questions (e.g. *Q228: What is platinum?*), whose implicit expected answer type is DEFINITION. The recognition of DEFINITION questions is based on matching a small set of question patterns against the user's question. Some of the patterns are:

(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 processing of questions asking for definitions does not use the expected answer type, but it is rather based on the recognition of the <*phrase_to_define*> and its matching one of the definition answer patterns. Some of the answer patterns are:

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

Examples of questions asking for definitions are illustrated in Table 3. These questions were tested during the TREC-9 evaluations.

Q228: What is platinum?
Q239: Who is Barbara Jordan?
Q358: What is a meerkat?
Q710: What is the definition of hazmat?

Table 3: Questions asking for definitions.

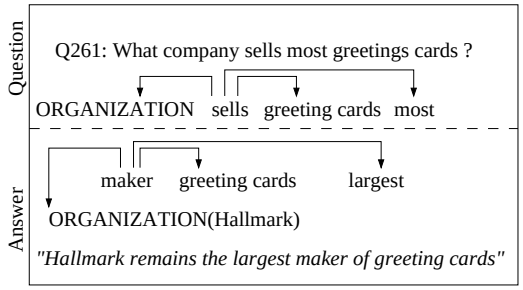
The other exception is represented by the case when the number of specializations of the question word that determines the expected answer type surpasses its number of instances. For example, the concept *planet* has *Mars*, *Pluto*, *Jupiter* and other stars as specializations, and fewer instances in the corresponding WordNet hierarchy (e.g. *asteroid*, *morning star*). In this case, the concept PLANET itself becomes an expected answer type. In the same way, the expected answer type of question *Q581: What flower did Vincent Van Gogh paint?* is FLOWER, with such sub-categories as *rose*, *sunflower* or *petunia*. This second exception is considered to have a *dynamic answer type*, determined by the ratio (*Number_of_specializations* /*Number_of_instances*) computed on WordNet hierarchies.

3. RETRIEVAL WITH FEEDBACKS

Finding the answer to a natural language question involves not only knowing *what* to look for (i.e. the expected answer type), but also *where* to look for the answer. The question is expressed with words that can be used in forming a query for an IR system, that returns sets of text passages, or *paragraphs* where the keywords and concepts of the expected answer type are found. The parse tree of the question indicates also the dependencies between the question words, thus imposes an order on the list of keywords that are used for retrieval. This ordered list can be used to take advantage of a paragraph retrieval implementation that employs the *SMART* IR engine [12]. The well-known disadvantage of boolean retrieval can be tackled by dropping some of the keywords when too few paragraphs are returned, or to add some keywords when too many meaningful paragraphs are found. This mechanism of adding/dropping keywords until either an acceptable number of paragraphs is retrieved or all the list of keywords has been processed generates the first feedback in the retrieval mechanism implemented in our Q/A system and illustrated in Figure 5. The minimal and

maximal number of paragraphs depends on the expected answer type, but generally does not exceed 500.

Paragraphs that do not contain the expected answer type are discarded, after which they are parsed and their dependencies are normalized to obtain the same structure as the question dependencies. The normalization is an assessment of the similarity between the question binary dependencies and the answer dependencies. The normalizations involving lexical, morphological or semantic knowledge expand the question and answer words to account for as many similarities as possible. The expansion is based on WordNet paths between concepts from the question and answer, respectively. An example is:



We find an entailment between producing, or making and selling goods, derived from WordNet, since synset {*make, produce, create*} has the genus *manufacture*, defined in the gloss of its homomorphic nominalization as *"for sale"*. Therefore the semantic form of question *Q261* and its illustrated answer are similar.

When lexico-semantic normalizations are not possible, a second feedback loop is generated, replacing question keywords with some of their alternations and searching for new relevant paragraphs. The selected alternations are based on the most frequent semantic links used in the paths of successful normalizations. The third feedback retrieval loop represented in Figure 5 takes place when multi-term semantic alternations are allowed in the normalizations. he normalizations used in this Q/A systems are presented in greater detail in [6]. Depending on the forms of linguistic knowledge that are employed, the alternations used in the feedbacks can be classified as:

∘ *Morphological Alternations.* Based on the specificity of the question keyword that has determined the expected answer type we enable all the morphological derivations that are accessible from WordNet. For example, in the case of question *Q209: Who invented the paper clip?* we allow all the morphological alternations of the verb *invented*. For this question, the verb was mapped into its nominalization *inventor*, which is in the subhierarchies of the answer type PERSON. Therefore, we passed to the retrieval engine the query:

QUERY(Q209):[*paper AND clip AND (invented OR inventor)*]

∘ *Lexical Alternations.* WordNet encodes a wealth of semantic information that is easily mined. Seven types of semantic relations span concepts, enabling the retrieval of synonyms and another semantically related terms. Such alternations improve the recall of the answer paragraphs. For example, in the case of question *Q221: Who killed Martin Luther King?*, by considering the synonym of *killer*, the noun *assassin*, the system retrieved paragraphs with the correct answer. Similarly, for the question *Q206: How far is the moon?*, since the adverb *far* is encoded in WordNet as being an attribute of *distance*, by adding this noun to the retrieval keywords, a correct answer is found.

∘ *Semantic Alternations.* Mining from WordNet semantic knowledge that is not always localized in the conceptual synset allows for semantic alternations. An example was used in the case of question *Q258: Where do lobsters like to leave?*. Since in WordNet the genus of the definition of the verb *prefer* is *liking better*, the query becomes:

QUERY(Q58):[*(lobster OR lobsters) AND (like OR prefer)*]

In this way the likelihood of retrieving the correct answer is greatly enhanced.

The main advantage of using feedbacks instead of expanding the keywords with all possible alternations comes from the fact that in WordNet there are many possible semantic paths between concepts, thus many expansions would not be necessary. Moreover, due to the properties of boolean retrieval, if the first retrieval loop would not be implemented, the overall precision of the Q/A system would be greatly affected, resulting in accuracy three times lower than the one obtained when the loops are active.

4. ANSWER EXTRACTION

4.1 A Machine Learning Approach

In our Q/A system the extraction of the text snippet where the answer of a question may lie is based on a perceptron model that was trained on the TREC-8 questions and applied to the TREC-9 questions during the evaluations. Our learning technique is based on the observation that the results of multiple feedback loops is always a set of paragraphs, in which at least one paragraph may contain the correct answer. Typically, the cardinality of the set of paragraphs is between 500 and 3000. Any sorting algorithm, e.g. *quicksort* can order this set of paragraphs if a *comparison function* is provided. The goal of the TREC Q/A evaluations is to return five ordered text snippets that represent the most likely answers to a given question. Therefore we need to sort all the paragraphs and return the first five paragraphs from which the text snippets can be extracted.

To learn the comparison function, we have experimented with numerous possible features and obtained the best results for the following seven features:

1] rel_{SP} the number of question words matched in the same phrase as the concept of expected answer type;

2] rel_{SS} the number of question words matched in the same sentence as the concept of expected answer type;

3] rel_{FP}: a flag set to 1 if the concept of expected answer type is followed by a punctuation sign, and set to 0 otherwise;

4] rel_{OCTW}: the number of question words matches separated from the concept of expected answer type by at most three words and one comma;

5] rel_{SWS}: the number of question words occurring in the same order in the answer text as in the question;

6] rel_{DTW}: the average distance from the concept of expected answer type to any of the question word matches;

7] rel_{NMW}: the number of question words matched in the answer text.

To train the perceptron we considered pairs of candidate answers. In the training phase, one of the paragraphs always contains the correct answer whereas its opponent is a paragraph returned by the multi-feedback retrieval. Given the pair of paragraphs (P_1, P_2), we compute $\Delta rel_{SP} = rel_{SP}^{P_1} -$

$rel_{SP}^{P_2}$; $\Delta rel_{SS} = rel_{SS}^{P_1} - rel_{SS}^{P_2}$; $\Delta rel_{FP} = rel_{FP}^{P_1} - rel_{FP}^{P_2}$; $\Delta rel_{OCTW} = rel_{OCTW}^{P_1} - rel_{OCTW}^{P_2}$; $\Delta rel_{SWS} = rel_{SWS}^{P_1} - rel_{SWS}^{P_2}$; $\Delta rel_{DTW} = rel_{DTW}^{P_1} - rel_{DTW}^{P_2}$; and finally $\Delta rel_{NMW} = rel_{NMW}^{P_1} - rel_{NMW}^{P_2}$. The perceptron computes a relative comparison score, given by 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 793 TREC-9 questions. We obtained the following values for the seven weights: $w_{SWS} = 12.458484$; $w_{FP} = -4.411543$; $w_{OCTW} = 3.1648636$; $w_{SP} = 4.461322$; $w_{SS} = 22.148517$; $w_{NMW} = 42.286851$; $w_{DTW} = -49.972180$. The learned value of the threshold is -15.051848.

At the test phase, given any pair of paragraphs, when the value of the resulting rel_{pair} is positive, the comparison function selects the first paragraph, otherwise it choses the second one. In addition, we found that prior to the extraction, the ordering of the paragraphs has significant effect on the overall performance of the Q/A system. To order the paragraphs we used again a perceptron, but this time we employed only four features. The definition of these four features depends on the notion of *paragraph-window*, first defined in [10]. Paragraph-windows are determined by the need to consider separately each match of the same keyword in the same paragraph. For example, if we have a set of keyword {*k1, k2, k3, k4*} and in a paragraph *k1* and *k2* are matched each twice, whereas *k3* is matched only once, and *k4* is not matched, we are going to have four different windows, defined by the keywords: [*k1-match1, k2-match1, k3*], [*k1-match2,k2-match1, k3*], [*k1-match1, k2-match2, k3*], and [*k1-match2, k2-match2, k3*]. A window comprises all the text between the lowest positioned keyword in the window and the highest position keyword in the window. Figure 6 illustrates the four windows for our example.

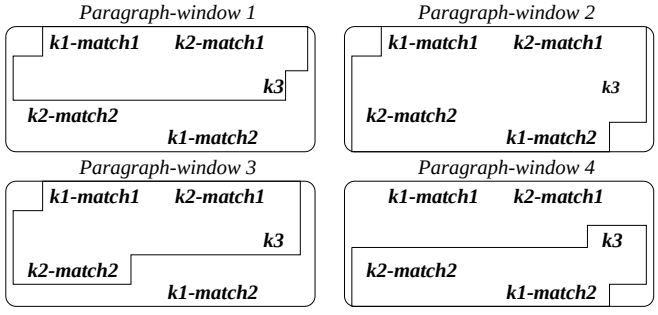


Figure 6: Four answer windows defined on the same paragraph.

For each paragraph window we compute the following scores:
1] rel_{SWS} computes the number of words from the question that are recognized in the same sequence in the current paragraph-window.
2] rel_{DAW} represents the number of words that separate the most distant keywords in the window.
3] rel_{NMW} computes the number of unmatched keywords. This measure is identical for all windows from the same paragraph, but varies for windows from different paragraphs.
The formula employed by the perceptron that learns how to order paragraphs by their paragraph-window scores is:

$$\begin{aligned} ord_{pair} \quad = \quad & q_{SWS} \times \Delta rel_{SWS} + q_{DAW} \times \Delta rel_{DAW} + \\ & + q_{NMW} \times \Delta rel_{NMW} + threshold \end{aligned}$$

We obtained the following values for the three weights: q_{SWS}=13.470984; q_{DAW}= -163.20379; q_{NMW}=-11.482971 and the threshold has the value 72.88456. At testing time, when the relative order measure ord_{pair} is positive, the first paragraph precedes the second one, otherwise their order is reversed.

4.2 Answer Caching

Before initiating the search for a question answers, we considered that it is very possible that the same question or a very similar one has been posed to the system before, and thus those results can be used again. To find such *cached questions*, we measure the similarity to the previously processed questions and when a reformulation is identified, we consider all *question reformulations* and their corresponding answers. To classify questions in reformulation groups, we successively built a similarity matrix $\mathcal{M}$. When a new question is posed, a new row and a new column is added to $\mathcal{M}$, containing flags signifying whether the new question is similar to any of the previous questions. Figure 7 represents the similarity matrix $\mathcal{M}$ for six questions that were successively posed to our Q/A system. Since question reformulations are transitive relations, if at step n questions Q_i and Q_j are found similar and Q_i already belongs to $\mathcal{R}$, a reformulation class previously discovered (i.e. a group of at least two similar questions), then question Q_j is also included in $\mathcal{R}$. Figure 7 illustrates the transitive closures for reformulations at each of the five steps from the succession of six questions. To be noted that at step 4 no new similarities were found , thus Q_5 is not found similar to Q_4 at this step. However, at step 5, since Q_6 is found similar to both Q_4 and Q_5, Q_4 results similar to all the other questions but Q_3.

	Q1	Q2	Q3	Q4	Q5	Q6	
Q1	0	1	0	1	0	0	
Q2	1	0	0	0	0	0	Step 1: {Q1, Q2}
Q3	0	0	0	0	0	0	Step 2: {Q1, Q2} {Q3}
Q4	1	0	0	0	0	1	Step 3: {Q1, Q2, Q4} {Q3}
Q5	0	0	0	0	0	1	Step 4: {Q1, Q2, Q4} {Q3} {Q5}
Q6	0	0	0	1	1	0	Step 5: {Q1, Q2, Q4, Q5, Q6} {Q3}

Figure 7: Building reformulation classes with a similarity matrix.

The similarity between two questions is computed by testing possible *Lexical_relation* between pairs of content words. Either identity between the words or one of the following three possible relaxations of *Lexical_relation* are allowed: (a) common morphological root (e.g. *owner* and *owns*, from question *Q742: Who is the owner of CNN?* and question

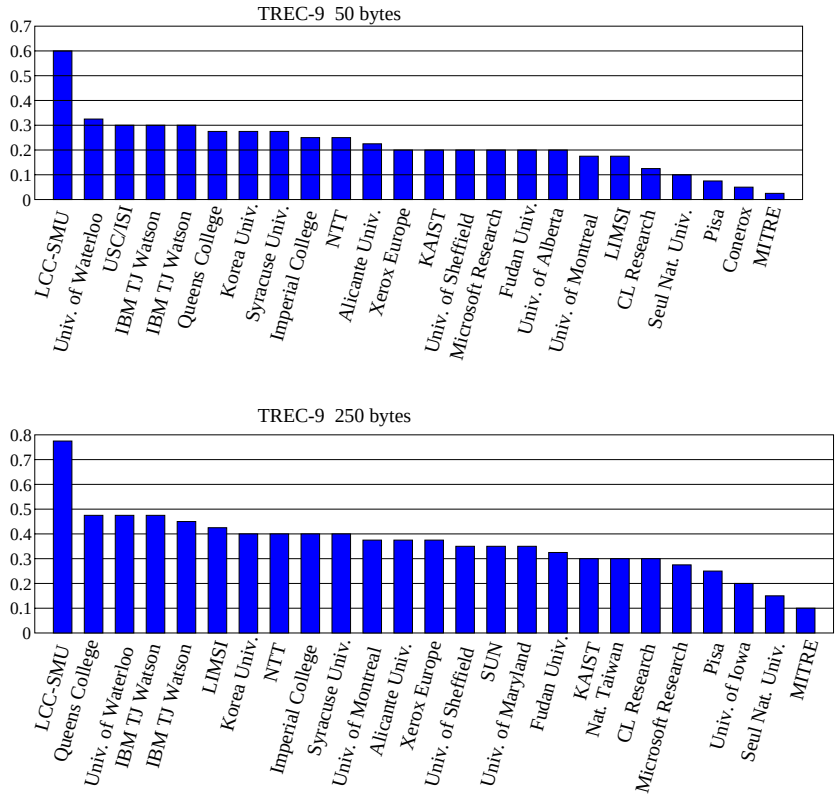


Figure 8: Results of the TREC-9 evaluations.

Q417: Who owns CNN? respectively); (b) WordNet synonyms (e.g. *gestation* and *pregnancy* from question *Q763: How long is human gestation?* and question *Q765: A normal human pregnancy lasts how many months?*, respectively) or (c) WordNet hypernyms (e.g. the verbs *erect* and *build* from question *Q814: When was Berlin's Brandenburg gate erected?* and question *Q397: When was the Brandenburg Gate in Berlin built?* respectively).

5. EVALUATION

To measure the performance of our Q/A system we start from the TREC-9 human-assessed evaluations of our submitted results. In TREC, for each question the performance was computed by the reciprocal value of the rank (RAR) of the highest-ranked correct answer given by the system. Given that only the first five answers were considered in the TREC evaluations, if the RAR is defined as $RAR = \frac{1}{rank_i}$ its value is 1 if the first answer is correct; 0.5 if the second answer was correct, but not the first one; 0.33 when the correct answer was on the third position; 0.25 if the fourth answer was correct; 0.2 when the fifth answer was correct and 0 if none of the first five answers were correct. The Mean Reciprocal Answer Rank (MRAR) is used to compute the overall performance of the systems participating in the TREC evaluation $MRAR = \frac{1}{n}(\sum_i^n \frac{1}{rank_i})$. In addition, TREC-9 imposed the constraint that an answer is considered correct only when the textual context from the document that contains it can account for it. When the human assessors were convinced this constraint was satisfied, they considered the RAR to be *strict*, otherwise, the RAR was considered *lenient*. Table 4 summarizes the MRARs provided by NIST for our Q/A system. Figure 8 shows the results of the TREC-9 Q/A evaluations. It shows that our system, marked `LCC-SMU` had much better performance than the other Q/A systems.

Table 4: NIST-evaluated performance

	MRAR *lenient*	MRAR *strict*
Short answer	0.599	0.580
Long answer	0.778	0.760

The first feature that we considered in our evaluation was the precision and coverage of our technique of finding the expected answer type. Table 5 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 TYPE 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 5. 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*.

Besides evaluating the coverage of our ANSWER TYPE

Table 7: Tests of answer extraction accuracy on the same test set

Test set	Score (No learning)	Score (With learning)	Nr. questions (learning worse)	Nr. questions (learning better)
Set1	0.887	0.892	3	3
Set6	0.370	0.580	3	9

Table 8: Cross-validation experiments for answer extraction. Set1 is the training set

Test Set	Score (No learning)	Score (With learning)	Nr. questions (learning worse)	Nr. questions (learning better)
Set2	0.925	0.938	1	2
Set3	0.703	0.697	9	8
Set4	0.306	0.396	5	8
Set5	0.361	0.587	2	9

Table 5: Results for the identification of the Expected Answer Type.

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

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

Table 6: Number of feedbacks on the TREC test data

	Average number	Maximal number
Loop 1	1.384	7
Loop 2	1.15	3
Loop 3	1.07	5

TAXONOMY we have been interested in analyzing the multi-feedback retrieval and its effect on the overall performance. Table 6 lists the quantitative analysis of the feedback loops. Loop 1 had not only the largest possible number of feedbacks but also the largest average number of feedback. Moreover, the overall average number of feedbacks indicate that they port little overhead to the Q/A system.

More interesting is the qualitative analysis of the many effects of the feedback loops on the Q/A evaluation. In the overall, the precision increase substantially when all loops were enabled. Individually, the effect of Loop 1 was an accuracy increase of over 40%, the effect of Loop 2 had an enhancement of more than 52% while Loop 3 produced an enhancement of only 8%. Table 9 lists also the combined effect of the feedbacks, showing that when all feedbacks are enabled, for short answers we obtained an MRAR of 0.568, an increase of 76%, whereas for long answers it was 0.737, which is an increase of 91%. Because we also used the answer caching technique, we gained more than 1% for short and almost 3% for long answers, obtaining the result listed in Table 4.

In addition, lexical alternations were used only for 129 questions whereas semantic alternations we employed for 175 questions of the total of 890 TREC questions.

To evaluate the answer extraction approach based on machine learning, we used separately 195 fact-seeking questions; 178 questions are selected from the TREC-8 and TREC-9 Q/A track evaluation questions, and 17 questions are real-world questions that were submitted to search engines. Table 10 illustrates the six sets of questions considered for the answer extraction evaluations.

Table 9: Effect of feedbacks on accuracy. L1=Loop 1; L2=Loop 2; L3=Loop 3.

L1	L2	L3	MRAR short	MRAR long
No	No	No	0.321	0.385
Yes	No	No	0.451	0.553
No	Yes	No	0.490	0.592
Yes	Yes	No	0.554	0.676
No	No	Yes	0.347	0.419
Yes	No	Yes	0.488	0.589
No	Yes	Yes	0.510	0.629
Yes	Yes	Yes	0.568	0.737

Table 10: Six test question sets

Set	Source	Nr. Questions
Set1	TREC-8 (subset1-trec8)	40
Set2	TREC-8 (subset2-trec8)	40
Set3	TREC-9 (subset1-trec9)	50
Set4	TREC-9 (subset2-trec9)	27
Set5	TREC-9 (subset3-trec9)	21
Set6	real questions (external-set)	17

First, we tested the accuracy of the extracted answers based on the perceptron models as opposed to those extracted by using manually assigned weights. Table 7 shows that the learned weights provide a better answer ranking as opposed to the ranking provided by manually-selected weights.

We also evaluated the exactness of the answer extraction by cross-validating the ranking learned on the first set on the other sets. The performance of the learned ranking mechanism, versus its empirical version is shown in Table 8. Consistently the learned weights allow for better ranking. Whenever the correct answer is not scored empirically on the first position, e.g. for Set4 and Set5, learning improves the precision score by 9.6% and 22.6% respectively. Conversely, when the returned answers are close to ideal, learning has a smaller impact on precision, yet it improves it for Set2 by 1.3%. Set3 is an exception because the precision score is slightly lower with learned weights. However, even

in this case, the ratio of the number of questions answered with improved accuracy over the number of questions answered with lower accuracy is close to 1 (9 questions receive worse RAR whereas 8 questions receive better RAR).

6. CONCLUSION

This paper has presented a Q/A system that combines a wide-coverage mechanism of identifying the expected answer type of open-domain natural language questions with a novel, multi-feedback retrieval scheme that brings forward paragraphs containing candidate answers. Evaluations indicate that the answer type taxonomy has over 90% precision and its coverage can be enhanced by relying on large, open-domain linguistic resources such as WordNet. A major contribution to the overall performance of the Q/A system is accounted by the three feedback loops implemented in the IR mechanism. The results show that when all these feedbacks are enabled an enhancement of almost 76% for short answers and 91% for long answers, respectively, is reached. In addition, a small increase is produced by relying on cached answers of similar questions. Our results so far indicate that the usage of feedbacks that produce alternations is significantly more efficient than multi-word indexing or annotations of large corpora with predicate-argument information. In addition, this paper presents a new method of extracting answers, based on machine learning techniques.

7. REFERENCES

[1] S. Abney, M. Collins, and A. Singhal. Answer extraction. In *Proceedings of the 6th Applied Natural Language Processing Conference (ANLP-2000)*, pages 296–301, Seattle, Washington, 2000.

[2] P. Cohen, R. Schrag, E. Jones, A. Pease, A. Lin, B. Starr, D. Easter, D. Gunning, and M. Burke. The DARPA High Performance Knowledge Bases Project. *Artificial Intelligence Magazine*, 19(4):25–49, 1998.

[3] M. Collins. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th Annual Meeting of the Association of Computational Linguistics (ACL-96)*, pages 184–191, Copenhagen, Denmark, 1996.

[4] A. Graesser and S. Gordon. *Question-Answering and the Organization of World Knowledge*, chapter Question-Answering and the Organization of World Knowledge. Lawrence Erlbaum, 1991.

[5] S. Harabagiu and S. Maiorano. Finding answers in large collections of texts: Paragraph indexing + abductive inference. In *AAAI Fall Symposium on Question Answering Systems*, pages 63–71, November 1999.

[6] S. Harabagiu, D. Moldovan, M. Paşca, R. Mihalcea, M. Surdeanu, R. Bunescu, R. Gîrju, V. Rus, and P. Morărescu. The role of lexico-semantic feedbacks in open-domain textual question/answering. In *39th Annual Meeting of the Association of Computational Linguistics (ACL-2001)*, Toulouse, France, 2001.

[7] B. Katz. From sentence processing to information access on the world wide web. In *AAAI Spring Symposium on Natural Language Processing for the World Wide Web*, Stanford, California, 1997.

[8] W. Lehnert. *The Process of Question Answering: a computer simulation of cognition*. Lawrence Erlbaum, 1978.

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

[10] D. Moldovan, S. Harabagiu, M. Paşca, R. Mihalcea, R. Goodrum, R. Gîrju, and V. Rus. LASSO: a tool for surfing the answer net. In *Proceedings of the 8th TExt Retrieval Conference (TREC-8)*, Gaithersburg, Maryland, 1999. NIST.

[11] J. Prager, D. Radev, E. Brown, A. Coden, and V. Samn. The use of predictive annotation for question answering in TREC8. In *Proceedings of the 8th TExt Retrieval Conference (TREC-8)*, Gaithersburg, Maryland, 1999. NIST.

[12] G. Salton, editor. *The SMART Retrieval System*. Prentice-Hall, New Jersey, 1969.

[13] R. Srihari and W. Li. Question answering supported by information extraction. In *Proceedings of the 8th TExt Retrieval Conference (TREC-8)*, Gaithersburg, Maryland, 1999. NIST.

[14] T. Winograd. Five lectures on artificial intelligence. *Linguistic Structures Processing. In Fundamental Studies in Computer Science*, 5:399–520, 1977.

[15] W. Woods and R. Kaplan. Lunar rocks in natural English: Explorations in natural language question answering. *Linguistic Structures Processing. In Fundamental Studies in Computer Science*, 5:521–569, 1977.