Analysing Definition Questions by Two Machine Learning Approaches

Carmen Martínez and A. López López
Instituto Nacional de Astrofísica, Óptica y Electrónica
Luis Enrique Erro # 1
Santa María Tonanzintla, Puebla, 72840, México
carmen@inaoep.mx, allopez@inaoep.mx

Abstract

In automatic question answering, the identification of the correct target term (i.e. the term to define) in a definition question is critical since if the target term is not correctly identified, then all subsequent modules have no chance of providing relevant nuggets. In this paper, we present a method to tag a question sentence experimenting with two learning approaches: QTag and Hidden Markov Model. We tested the methods in five collections of questions, PILOT, TREC 2003, TREC 2004, CLEF 2004 and CLEF 2005. We performed ten-fold cross validation for each collection and we also tested with all questions together. The best accuracy rates for each collection were obtained using QTag, but with all questions together the best accuracy rate is obtained using HMM.

1. Introduction

Question Answering (QA) is a computer-based activity that tries to improve the output generated by Information Retrieval (IR) systems, and involves searching large quantities of text and "understanding" both questions and textual passages, to the degree necessary to recommend a text fragment as an answer to a question.

Regarding the input of QA systems, according to [1] there are five sorts of questions:

1. Factual questions. The answer is a number, short phrase or sentence fragment obtained from one document (e.g. When was the telegraph invented?).

2. List questions. The answer is a list of an exact number of short phrases or sentence fragments from different documents (e.g. Name 20 countries that produce coffee).

3. Definition questions. The answer is a list of complementary short phrases or sentence fragments from different documents (e.g. What are nanoparticles?, Who was Christopher Reeve?).

4. Complex questions. The question is separated in sub-questions so, to answer the complex question, the sub-questions have to be answered first (e.g. How have thefts impacted on the safety of Russia's nuclear navy, and has the theft problem been increased or decreased over time? a) What specific instances of theft do we know about? . . . e) What is meant by nuclear navy?).

5. Speculative questions. To answer this kind of question, it is necessary some kind of reasoning (e.g. Is the airline industry in trouble?).

There are seven interrogative adverbs (*who, why, how, which, what, where, when*), from these only *what* and *who* can be interrogative adverbs for definition questions since they express a request for *a concise explanation of the meaning of a word, phrase, symbol or explanation of the nature of a person or thing.*

Please use the following format when citing this chapter:

Martínez, C., López, A.L., 2006, in IFIP International Federation for Information Processing, Volume 217, Artificial Intelligence in Theory and Practice, ed. M. Bramer, (Boston: Springer), pp. 277–284.

Who can be used to formulate both factual and definition questions. So, if a question is *who is the president of Mexico?*, this is not a definition question since just requires a name, but *who is Vicente Fox?*, demands an explanation about a specific person.

Usually, when we talk about a definition we mean a sentence or a paragraph. For instance, a definition of *nugget* would be *a solid lump of a precious metal (especially gold) as found in the earth.* But according to the current state of the art in definition question answering [2], the reply is a set of only sentence fragments (precisely called nuggets). So, for the example "nugget", the answer can be the following fragments: *a solid lump, precious metal, gold, earth.*

When evaluating systems answering definition questions, a set of terms are given by assessors, who developed the questions. Also, these topics are given already classified as *vital* (important) and *ok* or non vital (less important).

Nowadays, definition questions have drawn much attention [2]. Answering definition questions is different to answering factual questions, as we described above, since in definition questions, there are several vital and non vital nuggets. In contrast, in factual questions the answer is a unique number, short phrase, or sentence fragment. Two representative works to definition questions answering are: Hildebrandt et. al. [3] presented a multi-strategy approach using a database constructed offline with surface patterns, a Web-based dictionary, and an off-the-shelf document retriever. They employed a simple pattern-based parser using regular expressions to analyze the questions. On the other hand, Tsur [4] used text categorization and a biography learner to improve the task, i.e. definition question answering. Questions analysis is rather naive based on keywords, articles, determiners, capitalization, and name recognition.

For all definition question systems, the first module is target extraction, i.e. the term to define. However, some authors [5, 6] that present an analysis of their errors, found that they obtained poor efficacy because many errors can be traced back to problems with target extraction. If the target term is not correctly identified, then all subsequent modules have no chance of providing relevant nuggets. So, given the question, a key problem to resolve is to obtain the target term since this will be the term to define. For instance, in the following questions:

What are nanoparticles?
Who is Niels Bohr?
What is Friends of the Earth?
Who was Abraham in the Old Testament?

Nanoparticles, Niels Bohr, Friends of the Earth and Abraham are target terms. We can identify three different structures of questions: when the target is a single term, e.g. a noun, when the target is a named entity, and when the target term comes with some other words that are possibly its context.

The main idea to analyze the definition question and obtain the target term and additional information (context terms) is: the interrogative adverb and the verbal form are removed from each question. Then, we apply a named entity tagger, if the result is only one word or one named entity, then there is no choice, that is the target term. For the rest of the questions, we apply a Part-Of-Speech (POS) tagger. From this, the idea is to check if the question follows a previously found pattern that can immediately reveal the target and context terms. To achieve this, we have to tag previously the known sentences to obtain a training set and make a special purpose tagger, i.e. a *question sentence tagger.* The principal tags that we used are *V*, for terms that can be ignored, *T* for the target term, and *C* for context terms.

The paper is organized as follows: next section describes briefly the learning algorithms: Hidden Markov Model (HMM) and QTag; Section 3 presents the method to tag question sentences; Section 4 reports experimental results; finally, some conclusions and directions for future work are presented in Section 5.

2. Learning Algorithms

In this section, we describe the two Machine Learning approaches, Hidden Markov Model and QTag, that we applied to solve the problem.

2.1. Hidden Markov Model

A Hidden Markov Model (HMM), as Rabiner describes in [7], is a Markov chain, where each state generates an observation. An HMM is specified by a five-tuple (S,K,Π,A,B), where S is the set of states, K the output alphabet and Π, A, B are the probabilities for the initial state, state transitions, and symbol emission, respectively.

Given appropriate values of S,K,A,B, and Π, the HMM can be used as a generator to return an observation sequence

$O = O_1 O_2 \cdots O_T$

where each observation O_t is one of the symbols from B, and T is the number of observations in the sequence.

There are three basic questions that we want to know about an HMM:

1. Given the observation sequence $O = O_1 O_2 \cdots O_T$ and a model $\lambda = (A, B, \Pi)$, how do we efficiently compute $P(O|\lambda)$, the probability of the observation sequence, given the model?.
2. Given the observation sequences O, and the model λ, how do we choose a corresponding state sequence $Q = q_1 q_2 \cdots q_t$ which is optimal in some meaningful sense (i.e., best "explains" the observations)?
3. How do we adjust the model parameters λ to maximize $P(O|\lambda)$?

In question 1, given a model and a sequence of observations, how do we compute the probability that the observed sequence was produced by the model. Question 2 is intended to uncover the hidden part of the model, i.e., to find the "correct" state sequence. Question 3 points to the process to optimize the model parameters to best describe how a given observation sequence is generated.

2.2. Applying HMMs to POS tagging

HMMs can be used to POS tagging but for this task, parameters can not be randomly initialized, since this would leave the tagging task too unconstrained. The symbol emission probabilities is initialized using the method of Jelinek [8]:

$$b_{j.l} = \frac{b^{\star}_{j.l} C(w^l)}{\sum_{w^m} b^{\star}_{j.m} C(w^m)}$$

where the sum is over all words w^m in the dictionary, and

$$b^{\star}_{j.l} = \begin{cases} 0 & \text{if } t^j \text{ is not a part of speech allowed for } w^l \\ \frac{1}{T(w^l)} & \text{otherwise} \end{cases}$$

where $T(w^j)$ is the number of tags allowed for w^j.

2.3. QTag

QTag [9] is a robust probabilistic parts-of-speech tagger. This is a program that reads text and, for each token in the text, returns the part-of-speech (e.g. noun, verb, punctuation, etc). QTag was advantageous for our needs because we can create our own resource files for a different language or tagset, we simply supply a pre-tagged training corpus. The size of the training data is obviously important for the accuracy of the tagging procedure.

3. The Method to Tag Question Sentences

The process to obtain the target term is the following:

We remove the interrogative adverb (*who* or *what*) and the verbal form (*is, are* or *was*) from each question. For example, from the questions given above, we get:

nanoparticles?
Niels Bohr?
Friends of the Earth?
Abraham in the Old Testament?

Then, we apply a named entity tagger (LingPipe) [10]. For the same questions, we obtain the following:

nanoparticles?
< type="PERSON" Niels Bohr > ?
Friends of the < type="LOCATION" Earth > ?
Abraham in the Old < type="PERSON" Testament > ?

CC	Coordinating Conjunction
CD	Cardinal number
DT	Determiner
IN	Preposition or Subordinating conjunction
JJ	Adjective
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
’	’
,	,
”	”
”	”
(	(
)	)
?	end

Table 1. Subset of tags produced by MBT.

V	void
T	*target*
C	context
’	’
,	,
”	”
”	”
(	(
)	)
?	end

Table 2. Tags used by the Question Sentence Tagger.

If the result is a single word or a named entity (as the first and second examples), then that is the target term. For the rest of the questions, we apply a Part-Of-Speech (POS) tagger, in our case Memory Based Tagging (MBT) [11] that, by the way, has a better performance tagging English questions than QTag. Table 1 details the subset of tags obtained so far. For the examples that we are using to illustrate and two additional examples, we obtain:

Friends/NNPS of/IN the/DT Earth/NNP ?/.
Abraham/NNP in/IN the/DT Old/NNP Testament/NNP ?/.
Treasury/NNP Secretary/NNP
Robert/NNP Rubin/NNP ?/.
the/DT International/NNP Committee/NNP
of/IN the/DT Red/NNP Cross/NNP ?/.

By tagging the sentence with part-of-speech, we generalize and work thereafter with patterns of questions, rather than raw text. Named entities within a context are also processed in this way (as noun phrases) in order to identify simultaneously target (named entity) and context. For the examples, we keep the following sequences of tags:

NNPS IN DT NNP ?
NNP IN DT NNP NNP ?
NNP NNP NNP NNP ?
DT NNP NNP IN DT NNP NNP ?

Then we tagged these sequences of part-of-speech labels according to our needs to obtain a training set and reach a special purpose tagger, i.e. a *question sentence tagger*. This is done in two ways, using QTag and HMM. Table 2 shows tags used by the *question sentence tagger*.

For the previous examples, we have

NNPS/T IN/V DT/V NNP/C ?/?
NNP/T IN/V DT/V NNP/C NNP/C ?/?
NNP/C NNP/C NNP/T NNP/T ?/?
DT/V NNP/T NNP/T IN/V
DT/V NNP/C NNP/C ?/?

These examples are part of the training set. Now if we have a new question, Who is Akbar the Great?, we apply the previous process:

Who is Akbar the Great?
Akbar the Great?
Akbar the < type="ORGANIZATION" Great > ?
Akbar/NNP the/DT Great/NNP ?/.
NNP DT NNP ?

The last sequence (NNP DT NNP ?) is tagged by the *question sentence tagger* using QTag or HMM. The correct tags are: NNP/T DT/V NNP/C ?/?, since "Great" serves as a context helpful to focus the search for a definition of the target term "Akbar".

4. Experimental Setting

We used definition questions from five collections:

COLLECTION	Simple	Complex	Total
PILOT	17	8	25
TREC 2003	31	19	50
TREC 2004	36	29	65
CLEF 2004	65	25	90
CLEF 2005	26	24	50
Total	175	105	280

In this table, by "simple" we refer to the questions where the target term is a single word or named entity and "complex" when the target term comes with some other words that are possibly its context. The collection PILOT [2] contains questions used in the pilot evaluation of definition questions performed by NIST and AQUAINT program contractors. TREC 2003 and TREC 2004 are sets of definition questions used to evaluate Questions Answering systems in the Text REtrieval Conference [12] in 2003 and 2004 respectively. The collections CLEF 2004 and CLEF 2005 are questions obtained from the Cross Language Evaluation Forum [13] in 2004 and 2005.

As mentioned above, we developed also a *question sentence tagger* using Hidden Markov Models (HMM) and QTag. An HMM is specified by a five-tuple (S, K, Π, A, B), where S is the set of states (each state is a tag), K the output alphabet, Π the initial state probabilities, A the state transition probabilities and B the symbol emission probabilities. The values used when the collection is ALL, are:

S= { BEGIN, V, T, C, ', ', ", ", (,), ? }
K= { BEGIN, DT, NNP, NN, NNPS, NNS, JJ,
IN, CC, CD,', ', ",",(,), ? }
Π = { 1 0 0 0 0 0 0 0 0 0 0 }

The transition probabilities (A) are generated randomly and improved with the training examples and B is initialized using the method of Jelinek described in the section 2.2. In order to form the training set used by QTag, we tagged the sequences of part-of-speech labels with the tags shown in Table 2, those tags are the same of the set S in the HMM.

	Complex	Global
PILOT	62.50	88
TREC 2003	33.33	76
TREC 2004	31.03	69.23
CLEF 2004	84	95.56
CLEF 2005	91.66	96
Average	60.50	84.96

Table 3. Comparison of the accuracy rates of the Question Sentence Tagger using QTag

	Complex	Global
PILOT	50	84
TREC 2003	27.78	74
TREC 2004	24.14	66.15
CLEF 2004	80	94.44
CLEF 2005	83.33	92
Average	53.05	82.12

Table 4. Comparison of the accuracy rates of the Question Sentence Tagger using HMM

5. Results

We performed two different experiments. In the first experiment, we tested separately each collection of questions, Table 3 shows the accuracy rates using QTag and the Table 4 shows the accuracy rates using HMM. In all tests, we made a ten-fold cross validation and the results are the average of five runs.

From the first experiment, we can observe that QTag performs better than HMM on the questions of interest, possibly because that is the kind of processing it was designed for. On the other hand, HMM performs poorly, caused by the small size of the training sets.

In the second experiment, we joined four collections of questions, PILOT, TREC 2003, TREC 2004, CLEF 2004 to form the collection that we called ALL. The collection ALL_1 contains the questions from the five collections. The collection ALL can be used as baseline since we can test if our method improves its performance when the training set increases. Table 5 shows the accuracy rates using QTag and the Table 6 displays the accuracy rates using HMM. Also we performed a ten-fold cross validation for each test.

The results of the second experiments show that HMM behaves better than QTag, from the beginning, with an increased training set. However, QTag is more sensitive to the increment in size of the training set, reflected in a higher percentage of improvement.

As one can observe, the results show that the method is feasible and delivers an acceptable level of accuracy for both approaches. As we increase the training set of question patterns, we expect to increase also the accuracy identifying target and context terms.

Our questions sentence tagger, in either version, had trouble tagging sentences with patterns under-represented. From very few examples, the pattern can not be learnt properly during training. Two instances of this kind of patterns are:

what is Micro Compact Car (MCC)?
NNP NN NN ?
what is the Order of the Solar Temple?
DT NNP IN DT NNP NNP ?

This problem will be overcome as the size of the training set increases.

	Complex	Global
ALL	38.75	78.70
ALL_1	51.43	81.80
% of Improvement	32.72	3.94

Table 5. Comparison of the accuracy rates of the Question Sentence Tagger using QTag

	Complex	Global
ALL	51.25	83.04
ALL_1	60	85
% of Improvement	17.07	2.36

Table 6. Comparison of the accuracy rates of the Question Sentence Tagger using HMM

6. Conclusions and Future Work

We have presented a method to identify the target term in an automatic, fast and flexible way. The method can be extended easily for new complex questions. As far as we know, definition question analysis has not been approached as a special tagging task, and given the results, seems very promising since questions are usually short and following certain patterns.

Moreover, with this method, we have additional information for the search of passages or documents for the answer, since the method identifies the target term along some other terms that are the context and valuable to refine the search for the definition.

Another advantage of our approach with a special purpose tagger is that we do not depend completely on a named entity tagger, specially in complex questions. For instance, the tagger can miss a named entity within a context, but the question tagger can identify target and context adequately.

Future work includes extending the corpus to train, and explore ensemble methods to improve the special purpose tagging. And finally, we have to integrate this method to the complete process of definition questions answering.

7. Acknowledgements

This work was partially supported by a CONACyT research grant U39957 and the scholarship 157233 for the first author.

References

[1] Dan Modolvan, Marius Pasca, Sanda Harabagiu, and Mihai Surdeanu. Performance Issues and Error Analysis in an Open-Domain Question Answering System. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 33–40, Philadelphia, July 2002.

[2] Ellen M. Voorhees. Evaluating Answers to Definition Questions. *NIST*, pages 1–3, 2003.

[3] Wesley Hildebrandt, Boris Katz, and Jimmy Lin. Answering Definition Question Using Multiple Knowledge Sources. In *Proceedings of HLT/NAACL*, pages 49–56, Boston, 2004.

[4] Oren Tsur. Definitional Question-Anwering Using Trainable Text Classifiers. Master's thesis, Institute of Logic Language and Computation, University of Amsterdam, 2003.

[5] S. Harabagiu and F. Lacatusu. Strategies for Advanced Question Answering. In *Proceedings of the Workshop on Pragmatics of Question Answering at HLT-NAACL*, pages 1–9, 2004.

[6] Jinxi Xu, Ana Licuanan, and Ralph Weischedel. TREC 2003 QA at BBN: Answering Definitional Questions. In *The Twelfth Text Retrieval Conference (TREC 2003)*, pages 28–35, 2003.

[7] Lawrence Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In *Proc. IEEE*, volume 77, pages 257–286, 1989.

[8] Christopher D. Manning and Hinrich Schutze. *Foundations of Statistical Natural Languaje Processing*. The MIT Press Cambridge, Massachusetts, London, England, 1999.

[9] Oliver Manson. Qtag-A portable probabilistic tagger. *Corpus Research, The University of Birmingham, U.K.*, 1997.

[10] http://www.alias-i.com/lingpipe/index.html.

[11] Walter Daelemans, Jakub Zavrel, Peter Berck, and Steven Gillis. MBT: A Memory-Based Part of Speech Tagger-Generator. In *Proceedings of the Fourth Workshop on Very Large Corpora, Copenhagen, Denmark,*, pages 14–27, 1996.

[12] http://trec.nist.gov/.

[13] http://www.clef-campaign.org/.