Off-line answer extraction for Question Answering
Mur, Jori
CHAPTER 1

INTRODUCTION

1.1 QUESTION ANSWERING: MOTIVATION AND BACKGROUND

In this age of growing availability of digital information the development of tools to search through an abundant supply of information is crucial. Well-known are information retrieval systems, of which online search engines, such as Google, are the most widely used examples. Typing in a few keywords results in a list of links to relevant documents.

There are, however, situations imaginable which require a more intuitive and natural approach to providing information. For example, if a client of a bank wants to know how to open a bank account or a traveller wants to know if he still can cancel his flight. People with questions like these are not looking for information in general, they want a direct, clear answer to a specific question. An adequate response would typically be a phrase or a couple of sentences instead of a full document. In these cases an online question-answering application could avail both the client and the company. For a company an online question-answering application is cheaper than a help desk while at the same time the client benefits from an online service which is available day and night. Furthermore, clients never need to wait on hold when using such systems, which in turn results in a higher level of client satisfaction.

Another situation in which a user would probably prefer a short answer to a complete document as reply to an information request will be as an application for cell phones. These days, cell phones provide a whole list of functions, of which access to Internet is already becoming a standard one. Surfing on the web on a cell phone requires some adjustments however, because of the small display. Instead of the usual search engines which return complete documents to information requests, an application that provides a short answer is to be preferred for a cell phone.

Corpus-based QUESTION ANSWERING SYSTEMS are designed to address this need for tools which provide users with small information bits based on natural language questions. Question Answering (QA) systems typically accept questions posed in natural language and return a precise answer to the user.
There is a major interest in the development of question-answering systems both from academic and from commercial groups. The well-known search engines Yahoo, Google and Microsoft Search all have launched question-answering systems,\(^1\) although they are mainly driven by the users themselves. Users can ask questions and gain points and reputation for answering other users’ questions. A rating mechanism makes sure that the best answer appears first. So it is rather a QA service than a system. The service is community based, i.e. questions are posed to the whole community and everyone who thinks he knows the answer can reply.

Another example of commercial question-answering is the software that Q-go\(^2\) develops for banks, insurance providers, and other companies. Questions are mapped to predefined model-questions. These model-questions are linked to an answer. Customers receive a direct answer to their questions from the company’s web-site. Another company using this approach is Ask Jeeves.\(^3\) For companies the advantage of this approach is that they can entirely control the output of the system. Only relevant questions are answered and the answers are fixed.

In the field of academia there is also much attention for question-answering, but as a research area it is certainly not new. Since the 1960s researchers have worked on QA-systems. The earliest systems can be described as natural language interfaces to databases (Green et al., 1961; Woods, 1973). For those systems databases are the information resource from which answers are extracted. User questions are automatically transformed into database queries. An example of this approach is START, on-line since December 1993. It has been developed by Boris Katz and his associates of the InfoLab Group at the MIT Computer Science and Artificial Intelligence Laboratory and it incorporates many databases found on the web (Katz, 1997). In the eighties many of the problems in practical QA became apparent and system development efforts were receding.

The rise of the World Wide Web in the late nineties has led to a renewed interest in the topic. A surge of academic activity was initiated by the Question Answering track of the Text REtrieval Conference (TREC) (Voorhees, 2000). The track started in 1999 (TREC-8) and provides a document collection and yearly a new question-answering set for the evaluation of question-answering systems with the goal to foster research on question-answering. Each year around 30 groups from academic, commercial, and governmental institutions participate in this evaluation run to compare the performance

1.1. Question Answering: motivation and background

The TREC QA track has focused solely on systems for English. Therefore the organisers of the Cross Language Evaluation Forum (CLEF) have set up a track for non-English monolingual and cross-language QA systems. Since 2003 five campaigns have been conducted, recording a constant increase in the number of participants and number of languages involved, i.e. in the last track of 2007 monolingual tasks were carried out for Bulgarian, German, Spanish, French, Italian, Dutch, Portuguese and Romanian (Giampiccolo et al., 2007).

Systems evaluated in these settings are typically based on information retrieval (IR) techniques. The QA systems use an IR system to select the top $n$ documents or paragraphs which match key words extracted from the question. Then candidate answers are found and ranked using among others information about the question type and the answer type.

In CLEF and TREC classification of questions is in part based on the techniques needed to answer them and in part on the type constraints that can be imposed on possible answers. The questions differ in their level of difficulty ranging from easy to very hard to answer. The most simple kind of questions are called **factoid questions**. Factoid questions are simple fact-based questions typically asking for a named entity. Named entities are names of persons, organisations, locations, expressions of times, quantities, monetary values, percentages, etc. We list a few example factoid questions here:

(1) a. When did the ferry the Estonia sink?
   b. How many moons orbit the planet Saturn?
   c. Who invented barbed wire?

Question that are considered harder to answer include **definition questions** such as (2-a), (2-b) and (2-c).

(2) a. What is UNICEF?
   b. Who is Tony Blair?
   c. What is religion?

For TREC the objective for these type of questions is to produce as many useful nuggets of information as possible. For example, the answer to (2-b) might include the following:
Born 6 May 1953
Prime Minister of the United Kingdom from 2 May 1997 to 27 June 2007
Appointed official Envoy of the Quartet on the Middle East
Elected Leader of the Labour Party in July 1994

For CLEF it is sufficient to return one relevant information nugget as answer. Still, these questions require special attention, because the system has to determine whether a given phrase really defines the question term or not. An answer such as second son of Leo would not be judged correct for question (2-b), although the fact is formally true. More details about definition questions can be found in 2.1.4.

Temporally restricted questions are also generally considered to be difficult questions to answer. These questions are restricted by a date or a period such as (3):

(3) Who was the prime minister of the United Kingdom from 2 May 1997 to 27 June 2007?

This question is hard to answer because one cannot simply look for the prime minister of the United Kingdom. The correct period should be taken into account too.

List questions are another class of questions. To answer such questions the system needs to assemble answers from information located in multiple documents. List questions can be classified into two subtypes: closed list questions such as (4-a) asking for a specific number of instances and open list questions such as (4-b) for which as many correct answers can be returned.

(4) a. What are the names of the two lovers from Verona separated by family issues in one of Shakespeare’s plays?
   b. List the names of chewing gum.

A fifth class of questions is the class of NIL questions. These questions do not have an answer in the text collection. Including these questions in the question set demands from the system that it determines whether an answer can be found at all. The idea is that a good QA system should know when a given question cannot be answered.

For several years now the TREC QA track has included follow-up questions. In 2007 they were also introduced by CLEF. These types of questions are grouped in topics, consisting of a set of questions such as in (5). Answering non-initial questions may require information from previous questions or answers to previous questions. In TREC descriptive topics are explicitly provided, whereas in CLEF only a numeric
1.1. Question Answering: motivation and background

topic ID is given.

(5) a. When was Napoleon born?  
b. Which title was introduced by him?  
c. Who were his parents?

**HOW-QUESTIONS** are sometimes included in the TREC and CLEF question sets, most of the time asking how somebody died (6-a). However, answers to these kind of questions are typically difficult to evaluate. Answers can be full sentences describing how things have happened (6-b), but also a single term such as in (6-c).

(6) a. How did Jimi Hendrix die?  
b. He had spent the night with his German girlfriend, Monika Dannemann, and likely died in bed after drinking wine and taking nine Vesperax sleeping pills, then asphyxiating on his own vomit.  
c. Alone.

Other question types such as questions based on nominal adjuncts (In CLEF called **EMBEDDED QUESTIONS**) such as (7-a), **YES/NO QUESTIONS** (7-b) and **WHY-QUESTIONS** (7-c) are typically not considered in the tracks of TREC and CLEF.

(7) a. When did the king who succeeded Queen Victoria die?  
b. Did people in the time of Christopher Columbus believe that the earth was flat?  
c. Why was China awarded with hosting the 2008 Olympics?

In section 2.1.4 we discuss the question set we use in our experiments. There one can find more information and examples for the question types which we have just introduced here.

To be able to perform the task of answering questions successfully, it is important to automatically understand questions such as given above, and to be able to locate answers to these questions in a text collection. Full grammatical understanding of both the question and the text can be very useful for this task.

For instance, a full parser recognises that in coordinations like (8), *Karlsruher SC* is actually the subject of the VP *verloor daarin met* ....

---

4The following examples are taken from the project description ([http://www.let.rug.nl/~gosse/Imix/project_description.pdf](http://www.let.rug.nl/~gosse/Imix/project_description.pdf), dd. May 6, 2008.)
Karlsruher SC made it into the finals last season, and lost 1-0 to the just demoted Kaiserslautern.

Similarly, full processing recognises that the relative pronoun in (9), corresponds to the object of *gepubliceerd*, and that the relative clause modifies *werkgelegenheidscijfers* and not *VS*.

De markt kijkt vol spanning uit naar de werkgelegenheidscijfers in de VS die vanmiddag om half vier Nederlandse tijd worden gepubliceerd.

Verbs selecting a non-finite verbal complement may either impose a reading where a grammatical object functions as subject of the complement (as in (10-a), where the object *Jeltsin* is the subject of *de wet te ondertekenen*), or where a grammatical subject is also the subject of the complement (as in (10-b), where *Abd ar-Rahmaan Moenief* is the subject of *schrijven*).

To force Jeltsin to sign the law, the Federation council has to take a stand.

Abd ar-Rahmaan Moenief, Jordanië’s bekendste schrijver, heeft met het *Verhaal van een stad* een biografie van Amman proberen te schrijven.

Shallow processing is typically insufficient for recognising such LONG-DISTANCE DEPENDENCIES and CONTROL DEPENDENCIES.

A full dependency parse of the question in (11-a) immediately reveals that *WebTV Networks* is actually the (logical) object of the verb *opkopen*, in spite of the fact that *WebTV Networks* functions as grammatical subject. It should be obvious that such information is useful, given potential answer strings like (11-b).

Door welk bedrijf werd WebTV Networks in 1997 opgekocht?

English: By which company was WebTV Networks bought in 1997?
This example indicates that knowledge about the possible realizations of dependency relations (i.e. subjects are typically realized as door-modifiers in passive sentences) can help to make the performance of QA based on dependency relations more effective.

This observation is also supported by the fact that more and more developers of QA systems are including syntactic information in their system architectures (Amaral et al., 2007; Pérez-Coutiño et al., 2006; Jijkoun et al., 2006).

The research described in the following chapters was carried out in the framework of the IMIX project Question Answering for Dutch using Dependency Relations. The aim of the IMIX project is to show that accurate and robust dependency analysis can be used to boost the performance of a QA system. This project investigates the use of sophisticated linguistic knowledge and robust natural language processing for QA. One of the goals of the project was to build a question answering system for Dutch. We have built such a question-answering system, which makes full use of a dependency analysis based on syntactic processing. The result of full parsing is a complete dependency analysis of the question and of the document collection.

1.2 THIS THESIS

In the present thesis all experiments are performed with the above-mentioned QA system. The information resource in which it searches for answers is a set of newspaper articles. The system applies two retrieval techniques. Most questions are answered by the technique based on retrieving relevant paragraphs from a document collection using keywords from the question. Potential answers are identified and ranked. The other retrieval strategy is based upon the observation that certain answer types occur in fixed patterns, and the corpus can be searched off-line for these kind of answers. We call this OFF-LINE ANSWER EXTRACTION and it is the focus of this thesis.

\footnote{A description of the research program is available at http://www.let.rug.nl/gosse/Imix/summary.html, dd. May 6, 2008.}
Chapter 1. Introduction

1.2.1 Off-line answer extraction: motivation and background

Extraction of specific information from raw text, whether it be answers to questions, isa-relations or other kind of relations, is a well-known task in the field of information science. Research has been focused on information extraction both as task in itself and as sub-task of other applications.

As task in itself it was largely encouraged by a series of seven message understanding conferences (MUCs 1987-1998). For each MUC participating groups were given sample messages and instructions on the type of information to be extracted. Shortly before the conference, participants were given a set of test messages to be run through their system. The output of each participant’s system was evaluated and then the results could be compared (Grishman and Sundheim, 1996). Tasks involved the extraction of information about a specific class of events called a scenario ranging from terrorist attacks (Kaufman, 1991) to joint ventures (Kaufman, 1993), to management changes (Kaufman, 1995) as well as missile launches (Kaufman, 1998).

One of the first to focus on the extraction of one specific relation instead of a scenario was Hearst (1992). She used part-of-speech patterns to extract hyponym relations (e.g. A Bambara Ndang is a bow lute). Berland and Charniak (1999) have done the same thing for part-of relations (e.g. “wheel” from “car”).

There has been an increased interest, also in the field of QA, to extract information from web pages.

In some cases the World Wide Web is seen simply as a vast resource of information. Brin (1998) has extracted author-title pairs from a repository of 24 million web-pages. Pašca et al. (2006) have shown that a small set of only ten seed facts about birth dates can expand to one million facts after only two iterations with very high precision using generalised patterns on a text collection containing approximately 100 million Web documents of arbitrary quality. The generalised patterns are produced from basic extraction patterns constructed from the context of seed facts. Terms in the generalised patterns are replaced with corresponding classes of distributionally similar words in this way covering an exponential amount of basic patterns.

Pantel and Pennacchiotti (2006) use the web basically to achieve high recall scores. The authors use generic patterns to retrieve instances from a specific corpus. Then they search on the Web for strings matching reliable patterns instantiated with the instances found by the generic patterns on the specific corpus. The idea is that correct instances will fire more with reliable patterns than incorrect ones. However, it seems if it also possible to use the reliable patterns directly on the Web to find instances.
In other cases extraction techniques based on HTML structures are applied. Cucerzan and Agichtein (2005) have exploited the explicit tabular structures created by web document authors.

Large repositories of semantic relations enable other applications to access this information. In Sabou et al. (2005), for example, syntactic patterns were defined to extract information from documents on bio-informatics to build a domain ontology. Thelen and Riloff (2002) applied information extraction for the construction of semantic lexicons.

The focus in this thesis is on the application of information extraction within question-answering. Information extraction then becomes answer extraction. The modifying term “off-line” is added to differentiate the process we focus on here from the answer extraction step typically implemented in question-answering systems to extract the exact answer from text snippets after passage retrieval. We extract answers before the actual process of question-answering starts.

Certain answer types frequently follow the same fixed patterns in a text. For example, information about birth dates is often formulated as follows: <X WAS BORN IN YEAR>, <X, BORN IN YEAR>, or <X (YEAR<sub>i</sub>-YEAR<sub>j</sub>)>. One can define such patterns and then search for answers before the actual process of question-answering starts, extracting all birth dates encountered and storing them in a repository. When a birth-date question is asked during the question-answering process the system can easily look up the answer.

In recent years several studies have appeared describing experiments which show the benefits of extracting information from raw text off-line for the task of question-answering.

Mann (2002) has extracted hyponym relations for proper nouns (e.g. ‘Emma Thompson is an actress’). From these hyponym relations he has built a proper noun ontology to answer questions such as: ‘Who is the lead singer of the Led Zeppelin band?’. The answer should be an instance of the term ‘lead singer’ in the ontology. To extract the hyponym relations he has used simple part-of-speech patterns achieving a precision of 60%.

Other works more closely related to our goals include Fleischman et al. (2003) and Bernardi et al. (2003). They both extracted facts from unstructured text using surface patterns to answer function questions such as ‘Who is the mayor of Boston?’ and ‘Who is Jennifer Capriati?’. Fleischman et al. described a machine learning method for removing noise in the collected data and they showed that the QA system based on this approach outperforms an earlier state-of-the-art system. Bernardi et al.
combined the extraction of surface text patterns with WordNet-based filtering of name-
apposition pairs to increase precision. WordNet is a semantic lexicon which groups
English words into sets of synonyms called synsets, providing short general definitions,
and recording the various semantic relations between these synonym sets (Fellbaum,
1998). However, the authors found that WordNet-based filtering hurt recall more
than it helped precision, resulting in fewer questions being answered correctly when
the extracted and filtered information was deployed for QA. They argued that an
end-to-end state-of-the-art QA system with additional answer-finding strategies and
statistical candidate answer re-ranking is typically robust enough to cope with the
noisy tables created in the off-line answer module.

Girju et al. (2003) presented a method to semi-automatically discover part-whole
part-of-speech patterns. Since these patterns can be ambiguous (‘NP\textsubscript{1} has NP\textsubscript{2}’
matches ‘Kate has green eyes’ which is a MERONYMIC RELATION, but also ‘Kate has a
cat’ which is a POSSESSION RELATION) semantic constraints were added. The targeted
part-whole relations were detected with a precision of 83%. The authors stated that
understanding part-whole relations allows QA systems to address questions such as
‘What are the components of X?’ and ‘What is X made of?’ illustrating this with an
elaborate example for the question ‘What does the AH-64A Apache helicopter consist
of?’.

Ravichandran and Hovy (2002) developed a bootstrapping answer extraction tech-
nique. Some seed words were fed into a search engine. Patterns were then automatic-
ally extracted from the documents returned and standardised. The simplicity of their
method has the advantage that it can easily be adapted to new languages and new do-
 mains. A drawback of this simplicity is that the patterns cannot handle long-distance
dependencies. This is particular problematic for systems based on small corpora since
they contain fewer candidate answers for a given question which makes the chances
smaller to find an instance of a pattern. The results of their experiments support this
claim. They performed two experiments: using the TREC-10 question set answers
were extracted from the TREC-10 corpus and from the web. The web results easily
outperformed the TREC results.

Lita and Carbonell (2004) introduced an unsupervised algorithm that acquires
answers off-line while at the same time improving the set of extraction patterns. In
their experiments up to 2000 new relations of who-verb types (e.g. who-invented, who-
owns, who-founded etc.) were extracted from a text collection of several gigabytes
starting with only one seed pattern for each type. However, during the task-based
evaluation in a QA-system the authors used the extracted relations only to train the
answer extraction step after document retrieval. It remains unclear why the extracted relations were not used directly as an answer pool in the question-answering process. Furthermore, the recall of the patterns discovered during the unsupervised learning stage turned out to be too low.

Tjong Kim Sang et al. (2005) compared two methods for extracting information to be used in an off-line approach for answering questions in a medical domain: one based on the structure of the lay-out of the text and one based on syntactic structures obtained by parsing the text. When evaluated in isolation both techniques obtain good precision scores. However, within a question-answering environment, the overall performances of the systems were disappointing. The main problem was the lack of coverage of the extracted tables. The authors stated that off-line information extraction is essential for achieving a real-time question-answering system. The layout-based extraction approach has a limited application because it requires semi-structured data. The parsing approach does not share this constraint, so that it therefore is more promising.

Arguments for applying off-line answer extraction have typically been in terms of speeding up the process of question answering. For example, Fleischman et al. (2003) claim that although information retrieval systems are rather successful at managing the vast amount of data available, the exactness required of QA systems often makes them too slow for practical use. Extracting answers off-line and storing them for easy access can make the process of question-answering orders of magnitudes faster. Especially in the case that one wants to use the World Wide Web as information resource, the benefits of speeding up question answering by using off-line answer extraction become clear.

Yet we claim that besides speed there are more reasons for choosing to use off-line answer extraction in addition to online answer extraction, for off-line answer extraction opens up the possibility to use different techniques, that can take advantage of different things, that otherwise would remain unfeasible.

The most significant difference is that the IR engine, a common component of online QA systems, can be omitted. Off-line we extract answers from the entire corpus, rather than just from the passages retrieved by the IR engine. In this way we avoid that answers are not found because the IR engine selected the wrong passages, making the answer extraction futile.

Furthermore, under the assumption that an incorrect fact is less frequent than a correct fact gathering facts together off-line opens up the possibility of filtering out noise using frequency counts. In addition, answers are classified into categories off-line.
Chapter 1. Introduction

During the question answering process the system only has to select the correct table to find a complete set of answers of the correct type. Experiments show indeed that we can achieve high precision scores with off-line answer extraction (Mur, 2005).

1.2.2 Research questions and claims

Much recent work in QA has focused on systems that extract answers from large bodies of text using simple lexico-syntactic patterns. These studies indicate one major problem associated with using patterns to extract semantic information from text. The patterns yield only a small subset of the information present in a text collection, i.e. the problem of low recall. The recall problem can be addressed by taking larger text collections as an information resource, for example the World Wide Web (Brill et al., 2001) or by developing more surface patterns (Soubbotin and Soubbotin, 2002).

However, the aim of the present thesis is to address the recall problem by using extraction methods that are linguistically more sophisticated than surface pattern matching. In section 1.1 we already argued that knowledge about the possible realisations of dependency relations can help to make the performance of QA more effective.

We can use this knowledge even further for more sophisticated techniques such as coreference resolution. In some cases more than one sentence should be taken into account to infer the answer. Reference is frequently used in all kind of expressions. We believe that recall of off-line answer extraction can be improved by applying techniques based on deep syntactic analysis of the corpus.

Specifically, we will try to answer the following questions:

• How can we increase the coverage of off-line answer extraction techniques without loss of precision?

• What can we achieve by using syntactic information for off-line answer extraction?

To answer these questions we have built an answer extraction module which is part of a Dutch state-of-the-art question-answering system. Even though the system we use is developed for Dutch, we believe that results of our experiments also apply to other languages.

Syntactic information is provided by a wide-coverage dependency parser for Dutch. Parsing the whole text collection opens up the possibility to define extraction patterns based on dependency relations.

The off-line answer extraction module is evaluated according to the number of facts it extracts. This requires a manual evaluation similar to Fleischman et al. (2003). Since
we are also interested in measuring the performance of this module as a sub-part of a questions answering system we also use the number of questions answered correctly as a performance indicator.

The major conclusions of our experiments are the following:

1. Answers to questions in a natural language corpus are often formulated in such complex ways that simple surface patterns are not flexible enough to extract them. More sophisticated extraction techniques based on dependency parsing can significantly improve the performance of off-line answer extraction.

2. Adding coreference information can improve the recall of off-line answer extraction without loss of precision.

3. For off-line answer extraction, low precision will not hurt the performance of question answering, while high recall will benefit it.

4. Off-line answer extraction is a useful addition to a question answering system, not only because it speeds up the process, but also because the whole corpus can be searched, thus avoiding possible errors in the IR component.

5. Learning patterns for answer extraction based on bootstrapping techniques is only feasible if the correct answers are formulated in a consistent way, which is often not the case.

6. The MUC score is a clear and intuitive score for evaluating coreference resolution systems.

1.2.3 Chapter overview

In this introductory chapter we presented the task of question-answering and the different approaches to this task. We argued that full grammatical understanding of both the question and the text can be very useful. This thesis focuses on off-line answer extraction, a separate module that can be built into a complete QA system. We believe that including this module helps to improve the results of a QA system.

In the current thesis we will investigate the role of off-line answer extraction within a QA system. More specifically, we want to examine how the coverage can be increased and how we can use syntactic information to improve the performance of this module.

In chapter 2 we perform a small experiment in which a state-of-the-art question-answering system is enhanced with an off-line answer extraction module. This experiment not only reveals the problems that we want to address in this thesis, it also
Chapter 1. Introduction

offers us the possibility to introduce the framework in which most experiments in the following chapters are performed.

To compare surface-based extraction techniques with dependency based extraction techniques we develop two parallel sets of patterns, one based on surface structures, the other based on dependency relations. This experiment is described in chapter 3. The results of the experiment show that the use of dependency patterns has a positive effect, both on the performance of the extraction task as well as on the question-answering task. Furthermore, we introduce equivalence rules. Added to the basic extraction patterns to account for syntactic variation they increase recall a lot. We also present the $d$-score, which computes the extent to which the dependency structure of question and answer match so as to take into account crucial modifiers in the question that otherwise would be ignored.

Chapter 4 presents a Dutch coreference resolution system, which makes extensive use of dependency relations. This system has been integrated into an answer extraction module. We evaluate the extended module on the extraction task as well as on the task of question-answering. We demonstrate that performance of both tasks increases using the information provided by the resolution system.

In chapter 5 we automatically learn patterns to extract facts by applying a bootstrapping technique. With these experiments we show that for the benefit of off-line QA recall is at least equally important as precision. Storing the facts off-line lets us use frequency counts to filter out incorrect facts afterwards, therefore we do not need to focus on precision during the extraction process. It is of greater importance that we extract at least the correct answer.

Finally, we summarise our findings in chapter 6, where we also give suggestions for future work.