Relation Extraction for Open and Closed Domain Question Answering

Gosse Bouma, Ismail Fahmi, and Jori Mur

Abstract One of the most accurate methods in Question Answering uses off-line information extraction to find answers for frequently asked questions. It requires automatic extraction from text of all relation instances for relations that users frequently ask for. In this chapter, we present two methods for learning relation instances for relations relevant in a closed and open domain (medical) question answering system. Both methods try to learn automatically dependency paths that typically connect two arguments of a given relation. The first (lightly supervised) method starts from a seed list of argument instances, and extracts dependency paths from all sentences in which a seed pair occurs. This method works well for large text collections and for seeds which are easily identified, such as named entities, and is well-suited for open domain question answering. In a second experiment, we concentrate on medical relation extraction for the question answering module of the IMIX system. The IMIX corpus is relatively small and relation instances may contain complex noun phrases that do not occur frequently in the exact same form in the corpus. In this case, learning from annotated data is necessary. We show that dependency patterns enriched with semantic concept labels give accurate results for relations that are relevant for a medical question answering system. Both methods improve the performance of the Dutch question answering system *Joost*.

Gosse Bouma
University of Groningen, Groningen, The Netherlands, e-mail: g.bouma@rug.nl

Ismail Fahmi
Gresnews Media, Amsterdam, The Netherlands e-mail: ismail.fahmi@gmail.com

Jori Mur
De Rode Planeet, Zuidhorn, The Netherlands e-mail: jori.mur@gmail.com

1 Introduction

In addition, work on speech recognition within IMIX (carried out in the NORISC project) is reported in (H
”am
”al
”ainen et al, 2005) and (Han et al, 2005).

Question answering is the task of finding answers to user questions in (large) text collections. Most QA systems use a technique where questions are analyzed to determine the expected answer type, passage retrieval is used to retrieve the most relevant text fragments from the text collection, and various NLP techniques are used to identify and rank phrases within the retrieved text that potentially answer the question and that match the expected answer type. An alternative method searches the corpus beforehand for answers to frequently asked question types. It requires that relation extraction patters are constructed (manually or automatically) that extract tuples of pairs instantiating a particular relation from the corpus. For instance, if questions about locations of museums are frequent, one can design patterns that would extract the tuple ⟨*Uffizi, Florence*⟩ from the sentence *Today the Uffizi is one of the most popular tourist attractions of Florence*. For those questions for which we developed relation extraction patterns, our open-domain Dutch question answering system *Joost* tends to give better results than using passage retrieval. Bouma et al (2005) show that on questions from CLEF 2003 - 2005, questions answered by consulting tables of previously extracted relation pairs achieve a mean reciprocal rank between 0.74 and 0.88, whereas questions answered by means of passage retrieval and answer extraction from text snippets results in a mean reciprocal rank between 0.53 and 0.62. This suggests that in general it is worthwhile to invest in relation extraction for question answering.

Much research on automatic learning of relation extraction patterns has concentrated on learning surface strings (i.e. sequences of words in the immediate context of the arguments of the relation) as extraction patterns. Such patterns can be found in unparsed corpora, and can be used to extract instance pairs from the web. Alternatively, one may learn dependency patterns. Dependency patterns abstract from many aspects of surface word order, and focus on those aspects of grammatical structure that seem most relevant for relation extraction. Below, we concentrate on the latter approach. For a language such as Dutch, which exhibits more word order variation than English, the fact that dependency patterns abstract over surface word order is important. Another reason for adopting dependency patterns is the fact that our question-answering system *Joost* (Bouma et al, 2005) also works with dependency paths in all other components of the system. Most importantly, for information-retrieval and answer extraction, we rely on the fact that the full text collection in which the system can find answers is syntactically parsed. Thus, we have access to dependency information and no additional effort is required.

In this chapter we address two important issues relevant to relation extraction for QA. First, manual construction of extraction patterns is labour intensive, especially if many question types need to be dealt with. Thus, it becomes interesting to study

the effect of lightly supervised relation extraction methods, that try to learn extraction patterns by bootstrapping from a small set of seed pairs, representative for the relation that needs to be learned. Such methods work well given a large corpus, in which relation instance pairs (such as ⟨*Uffizi, Florence*⟩ for the *museum-location* relation) can be found frequently and in many different contexts. We show below that, for the purposes of QA, lightly supervised bootstrapping methods can improve the performance of a QA system, even if the accuracy of the automatically retrieved instance pairs is relatively low.

For closed-domain, medical, QA it holds that the number of questions types is limited. Most questions will be about *definitions, causes, symptoms* and *treatments*. This suggests relation extraction could be very effective for a medical QA system. A problem for domain specific relation extraction, however, is the fact that corpora tend to be smaller than those used for open-domain QA, and thus there are fewer highly frequent instance pairs. Second, whereas relation extraction for open domain QA has concentrated on learning relations between named entities, the arguments of medical relations are often complex noun phrases that are subject to more grammatical variation than named entities. This is an additional factor that reduces the frequency of easily identifiable instance pairs. Therefore, most systems for relation extraction in the medical domain have made use of two additional resources to make the task feasible. First, a thesaurus (such as UMLS (Bodenreider, 2004)) is used to identify relevant concepts in text. Second, instead of learning from seeds, extraction patterns are learned on the basis of an annotated text corpus. In an annotated corpus, examples of the relation to be learned are marked, and the relation extraction system uses these positive examples to learn which grammatical patterns are typical for the relation.

Most work on medical QA has been done for English. For a Dutch medical QA system, relevant resources are less easy to obtain. Below, we show that UMLS can also be used for concept labeling of Dutch text. Simple string matching, matching of stems, and matching of automatically translated phrases allows a substantional number of Dutch medical terms to be matched with their English counterpart in UMLS. Second, we perform relation extraction experiments using the IMIX corpus developed by the IMIX ROLAQUAD team. This is a 60K word corpus of Dutch medical text annotated with semantic concepts and relations. We show that a system that extracts instance pairs by means of relation extraction patterns, and filters the results by imposing constraints on the semantic classes to which the arguments must belong, gives accurate results.

In the next section, we discuss previous work on relation extraction for open domain QA and for the medical domain, and we further motivate our choice for learning dependency pattterns for relation extraction. In section 3, we briefly introduce our QA system *Joost* and the construction of dependency patterns from parsed corpus data. In sections 4 and 5 we present our method for relation extraction in the context of open-domain QA, and for medical QA, respectively. We conclude with a discussion of our results.

2 Related Work

2.1 Relation extraction for open domain QA

Soubbotin and Soubbotin (2002) presented a question answering mechanism which uses predefined surface patterns to extract potential answer phrases. Inspired by the good results of this system in the TREC 2001 evaluation, Ravichandran and Hovy (2002), Fleischman et al (2003) and others investigated techniques for learning such extraction patterns automatically. In particular, these systems find answers to frequently asked question types by bootstrapping from a small list of seed pairs. Each sentence in which the seed appears leads to a potential extraction pattern (consisting, for instance, of the words intervening between the two arguments of the seed pair). The precision of a pattern is estimated by counting how often one finds the correct answer when one of the argument positions is filled in and the resulting string is submitted as a search query for a web search engine. For example, for the birthday relation, the person name can be supplied, and one can count how often the correct answer is found among all matching strings. Patterns yielding a high precision score are applied to find the answers to questions during the process of question answering. In our experiments aimed at extracting relations between named entities we use the same metric for selecting accurate extraction patterns.

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. Etzioni et al (2005) present an overview of their information extraction system KNOWITALL. Extraction patterns are found by instantiating generic rule templates with predicate labels such as *actor* and *city*. Using these patterns a set of seed instances is extracted with which new patterns can be found and evaluated. KNOWITALL introduces the use of a form of point-wise mutual information between the patterns and the extracted terms which is estimated from Web search engine hit counts to evaluate the extracted facts.

Pantel and Pennacchiotti (2006) describe ESPRESSO, a minimally supervised bootstrapping algorithm that takes as input a few seed instances and iteratively learns surface patterns to extract more instances. Besides a text corpus of 6.3 million words, the Web is used to increase recall. To evaluate patterns as well as extracted instance pairs, the authors calculate an association score between a pattern and instances based on point-wise mutual information. In section 4, we report briefly on an experiment in which we used the ESPRESSO method to extract relation instance pairs from parsed data.

2.2 Biomedical relation extraction

Relation extraction from biomedical text is an active research area. However, this research is restricted almost completely to English (e.g. medline abstracts), and tends to make heavy use of terminological resources such as MESH[1] and UMLS (Unified Medical Language System).[2] Rosario and Hearst (2004) observe, for instance, that "part of the reason for the success of [their relation extraction algorithms] is the use of a large domain-specific lexical hierarchy for generalization across classes of nouns". Leroy and Chen (2005) stress the importance of concept labeling, by observing that if the name of a gene and a disease cooccur, it is almost certain that there is a (causal) relation between the two.

When trying to apply similar techniques to languages other than English, one immediately runs into the problem that suitable terminological resources are lacking or have only limited coverage. At the same time, attempts at biomedical relation extraction without access to a terminological resource tend to give poor results. Tjong Kim Sang et al (2005), for instance, evaluate the performance of various relation extraction systems for Dutch in the context of a medical question-answering system, and conclude that both recall and precision is low. One of the reasons for low precision is the fact that these systems do not have access to concept labels.

In section 5, we concentrate on medical relation extraction for a language other than English. In particular, we address the issue of accurate concept labeling of Dutch medical terms and show that by combining the Dutch and English parts of UMLS reasonable coverage and accuracy can be achieved. Braun et al (2005) attempt to do full translation of Dutch medical terms into English on the basis of UMLS, for a cross-lingual information retrieval system, and find that the accuracy of automatic translation is low. Our task is different, in that we only need to assign a semantic concept label to a (Dutch) term, which does not always require a translation that would be useful for IR.

2.3 Using syntactic patterns

In contrast to the bootstrapping approaches discussed above we learn patterns based on dependency relations instead of surface patterns. Using dependency relations, we can simply define the extraction pattern as the shortest path between the two terms in a dependency graph. Surface patterns are typically harder to define in a natural and meaningful way.

Using syntactic (dependency) analysis for relation extraction has become increasingly popular in recent years (Bunescu and Mooney, 2005; Culotta and Sorensen, 2004; Zhao and Grishman, 2005). Most of this work relies heavily on annotated corpora, such as the ACE corpus, in which relations between named entities are marked

[1] http://www.ncbi.nlm.nih.gov/mesh

[2] http://umlsinfo.nlm.nih.gov

explicitly. In the medical domain, Rinaldi et al (2006) and Fundel et al (2007) use manually defined relation extraction patterns based on dependency trees, and Katrenko and Adriaans (2007) apply machine learning for learning medical extraction patterns. In section 4, we apply a lightly supervised approach for learning relations between named entities. We use fully parsed, but otherwise unannotated, data. In section 5, we use a corpus annotated with medical concepts and relations to learn dependency patterns between medical concepts.

One of the main reasons for adopting dependeny patterns is that it allows one to ignore intervening constituents and variations in word order that are not essential for the extraction pattern. For a language such as Dutch, this may be particularly important, as Dutch has a high degree of word order variation. The examples in (1), for instance, all contain a causal relation expressed by the phrase *wordt veroorzaakt door* (*is caused by*). If we have access to surface word order only, identification of a single extraction pattern from such data is practically impossible. Using dependency paths, on the other hand, one can extract a path linking the subject of the passive auxiliary *worden* to the object of the prepositional *door* modifier of the particple *veroorzaakt* from all these sentences.

(1) a. AIDS wordt veroorzaakt door het retrovirus HIV (AIDS *is caused by the retrovirus* HIV).
b. Nachtblindheid wordt meestal veroorzaakt door een tekort aan vitamine A (*Night blindness is usually caused by a lack of vitamin A*)
c. Echte griep of influenza is een ziekte die veroorzaakt wordt door het influenzavirus (*Real flu or influenza is a disease that is caused by the influenza virus*)
d. Buiktyfus is een geheel andere (darm) ziekte , die door Salmonella bacteriën wordt veroorzaakt (*Typhoid is a whole other (intestine) disease, which is caused by Salmonella bacteria*)
e. Brucellose bij mensen wordt met name door brucella melitensis veroorzaakt (*Brucellosis in humans is often caused by Brucella melitensis*)

The use of dependency patterns obtained from large amounts of automatically parsed data has recently also been explored for various lexical acquisition tasks, such as paraphrase learning or acquisition of taxonomic information. Lin and Pantel (2001), for instance, use 1 Gb of text parsed with Minipar (Lin, 2003), from which they extract 7M dependency paths and 200K unique paths, for learning paraphrases. Snow et al (2005) use a newswire corpus of 7M sentences, from which they extract 700K unique noun pairs, for learning hypernyms. McCarthy et al (2007) use the written portion of the British National Corpus (90M words), parsed with RASP Briscoe and Carroll (2002), to construct a thesaurus for learning predominant word senses. Padó and Lapata (2007), finally, use all of the 100M words from the BNC parsed with Minipar for a range of lexical semantic acquisition tasks. In the experiments below, we use an automatically parsed version of a Dutch newspaper corpus (80M words) and a medical corpus consisting of web pages, reference works, and Wikipedia (almost 3M words). In more recent experiments, discussed briefly in section 4, we have used a 700M word corpus.

3 Dependency Information for Question Answering and Relation Extraction

Alpino (Bouma et al, 2001; van Noord, 2006) is a wide-coverage, robust, parser for Dutch. Its grammar is designed following ideas of Head-driven Phrase Structure Grammar (Pollard and Sag, 1994), it uses a maximum-entropy model for statistical disambiguation, and coverage has been increased over the years by means of semi-automatic extension of the lexicon based on error-mining (van Noord, 2004). Efficiency is improved by using a part-of-speech tagger to filter out unlikely POS tags before parsing (Prins and van Noord, 2001), and by means of a technique which filters unlikely derivations based on statistics collected from automatically parsed corpora (van Noord, 2009).

Alpino is a crucial component of *Joost*, an open-domain question-answering system for Dutch (Bouma et al, 2005). Within the IMIX project, Joost was used as a QA module of the interactive, multimodal, medical QA system. We also used Joost to participate in the CLEF QA evaluation tasks, and achieved the best results for Dutch (Bouma et al, 2006).

Whereas most QA systems only use parsing to analyze the question and sometimes to analyze text snippets returned by the IR component, we used Alpino to parse the complete text collections used in the various QA systems (ranging from 2M to 110M words). The benefits are that syntactic information can be used to optimize the IR process (Tiedemann, 2005), and that off-line answer extraction can be based on dependency patterns. Jijkoun et al (2004) show, for instance, that both recall and precision of patterns for extracting answers off-line improve if patterns are dependency paths, instead of surface strings.

A succesfull component of many QA systems is the ability to answer questions not only by means of a method that extracts potential answers from passages returned by an information retrieval component, but also to answer questions using data that was collected by means of information extraction. For instance, if users ask frequently for the birth date of famous persons, one may use information extraction to locate all instances of the *person-birth date* relation in the corpus beforehand. It has been shown that using the results of either manually constructed (Soubbotin and Soubbotin, 2002; Bouma et al, 2006), or automatically created (Ravichandran and Hovy, 2002; Fleischman et al, 2003) tables with relation instances improves the accuracy of a QA system considerably. Our QA system incorporates the possibility to answer questions by means of table look-up, where the tables contain facts extracted by means of manually or automatically constructed extraction patterns. As we use parsed corpora for our QA system, extraction patterns are formulated in terms of grammatical dependency patterns.

For the information extraction experiments described below, we use dependency trees as produced by Alpino to extract dependency paths connecting two entities. In particular, given a pair of entities occurring in a single sentence, we extract from the dependency tree for that sentence the dependency pattern connecting the two entities. A dependency pattern in our implementation is the shortest path through the

tree connecting two nodes, where the nodes themselves are replaced by placeholders ARG1 and ARG2.

For example, for the sentences in (2), Alpino produces the dependency trees given in Figure 1.

(2) a. Begin volgend jaar treedt ook het Spaanse Telefónica tot Unisource toe (*Early next year, the Spanish Telefónica will also join Unisource*)
b. Een gebrek aan insuline leidt tot suikerziekte (*A shortage of insulin leads to diabetes*)

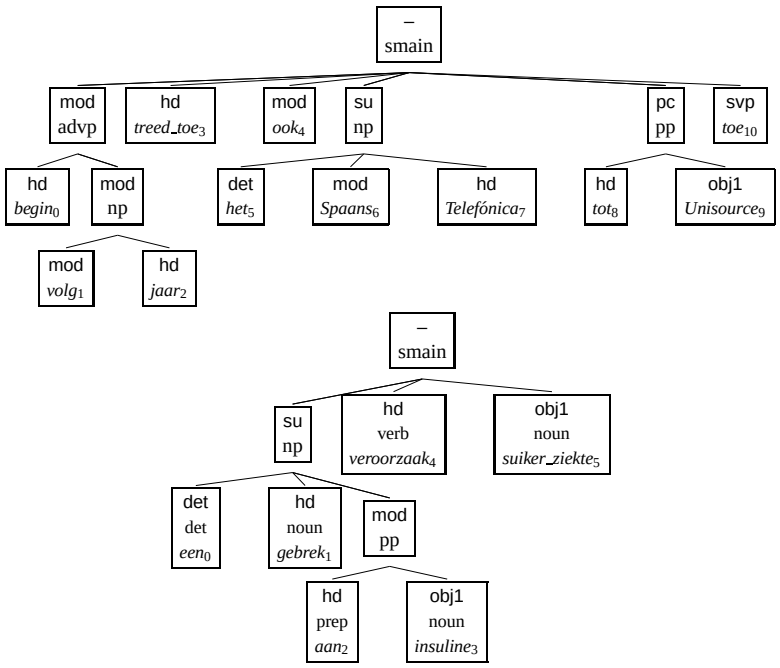

Fig. 1 Dependency tree for (2-a) and (2-b)

The dependency patterns connecting *Telefónica* and *Unisource* and *insuline* and *suikerziekte*, respectively, are:

(3) a. ARG1+*su* ← *treed_toe* → *pc*+*tot*+*obj1*+ARG2
b. ARG1+*obj1*+*aan*+*mod*+*gebrek* ← *leid* → *pc*+*tot*+*obj1*+ARG2

Given a dependency tree, the shortest path connecting two nodes is constructed by starting from one of the arguments, and going up in the tree until a constituent is reached that dominates the other node as well. Each time one has to go one node up, a string *rel*+*hd-root* is suffixed to the path expression, where *rel* is the dependency relation of the current node, and *hd-root* is the root form of the head sister node

(labeled with *hd*). The same is done for the second argument, but now the expression *hd-root+rel* is prefixed to the path expression. The head root of the minimal constituent that dominates both arguments is used as pivot expression. If one of the arguments is itself the head of a constituent dominating the other argument, this argument is itself the pivot, and one half of the pattern remains empty.

As pointed out above, one advantage of using dependency paths over patterns based on the surface string, is that dependency paths are able to deal with word order variation. Note that this is especially relevant for languages like Dutch or German, where there is considerable word order freedom, as illustrated by the (somewhat abbreviated) grammatical variants of (2-a) in (4).

(4) a. Ook Telefónica treedt begin volgend jaar tot Unisource toe
b. Ook Telefónica treedt begin volgend jaar toe tot Unisource
c. Telefónica treedt begin volgend jaar ook toe tot Unisource
d. Begin volgend jaar treedt Telefónica toe tot Unisource

For surface-based approaches, each word order variant may lead to a separate pattern, whereas our method extracts the same dependency path in each case. Another advantage is that dependency paths often capture more of the relevant context than surface patterns. Note, for instance, that the verb stem in (2-a) (*treedt*) precedes the subject, while a verbal particle (*toe*) follows the object. Surface based pattern extraction methods tend to concentrate on the string between the two arguments in a relation, and not always capture enough of the preceding or following context to obtain an accurate pattern. Finally, note that the preceding context contains an adverb, *ook*, and the name *Telefónica* is prefixed with a determiner and a modifier (*het Spaanse*), which most likely are not relevant for formulating an accurate pattern, and thus would have to be ignored somehow.

Stevenson and Greenwood (2009) compare various methods for using dependency tree information in pattern creation for IE. Methods extracting only *subject-verb-object* tuples have limited coverage, whereas methods extracting the minimal subtree containing both arguments suffer from lack of generality. Their *linked chain* method corresponds to our shortest path pattern extraction method, and performs well in an evaluation using the Wall Street Journal and biomedical data.

4 Relation extraction for open domain QA

In this section, we present a simple lightly supervised information extraction algorithm which operates on a parsed corpus to learn dependency patterns for extracting relation instances. Our primary goal is to use this system for off-line extraction of instance pairs that can be used to provide answers for frequently asked question types. Thus, we evaluate the results of the extraction process not only in terms of precision, but also by integrating them into the QA system *Joost*.

The algorithm takes as input seed pairs representing a particular question category. For example, for the category of *capital-of* questions we feed it ten *country-*

capital pairs like *Germany-Berlin*. From each sentence in which a seed pair occurs, we extract the dependency path connecting the two elements. Extraction patterns are selected by estimating the precision of each dependency path, and preserving only those paths that are above a given threshold.

The results of our experiments indicate that, for question answering, an extraction method that aims for high recall (possibly at the expense of precision) gives the best results. That is, the performance of our QA system improves most if we use high coverage, but sometimes extremely noisy, tables of relation instances.

4.1 Pattern induction

We start with a number of seed pairs. Extraction patterns are found by searching the corpus exhaustively for all sentences in which both arguments of a seed pair occur. The shortest dependency path between the two arguments is selected as extraction pattern. For instance, given a seed pair *Letland, Riga*, we construct the pattern (5-b) for sentence (5-a).

(5) a. Riga, de hoofdstad van Letland, is een mooie stad (*Riga, the capital of Latvia, is a beautiful city*)
b. ARG2 → *app*+*hoofdstad*+*mod*+*van*+ARG1

A given seed set typically gives rise to a large set of dependency patterns, especially if some of the seed pairs are frequent in the corpus. Not all dependency patterns are adequate as extraction patterns, however. Sentence (6-a) gives rise to the highly general pattern (6-b), for instance. Including this pattern as extraction pattern would mean that many false instance pairs are extracted from the corpus in the extraction stage.

(6) a. De in Riga in Letland geboren Leibowitz studeerde in Berlijn (*Leibowitz, born in Riga in Letland, studied in Berlin*)
b. ARG1 → *mod*+*in*+ARG2.

In the pattern-filtering stage, we filter unreliable patterns. To find the optimal balance between precision and recall for off-line answer extraction we perform three different bootstrapping experiments in which we vary the precision threshold for selecting patterns. The precision of a pattern is calculated following the method in Ravichandran and Hovy (2002). Instead of replacing both seed terms by a variable as in (5-b), we now replace only the answer term (i.e. of *what is the capital of... ? questions* with a variable:

(7) ANSWER → *app*+*hoofdstad*+*mod*+*van*+*Letland*

For each answer pattern obtained in this way, we count how many times the pattern occurs in the corpus, and we count how many times the variable ANSWER matches

with the correct answer term according to the seed list. The precision of each pattern is calculated by the formula

$$P = C_a/C_o$$

where C_o is the total number of times the pattern occurred in the corpus and C_a is the total number of times the pattern matched and ANSWER matched the correct answer term.

All patterns that occur at least two times in the corpus, and that have a precision score exceeding a set threshold τ_p are preserved for the instance extraction phase. The patterns that have passed the filter in the previous stage are matched against the parsed corpus to retrieve new relation instance pairs. After retrieval we select a random sample of one hundred instance pairs and manually evaluate it. If more than τ_f facts are found the iteration process is stopped, else all facts are used without any filtering as seeds again for the pattern-induction stage and the process repeats itself. In our experiments we have set τ_f to 5000.

4.2 Experiment

Experiments were performed using the CLEF corpus. This is a 80M word consisting of newspaper articles from 1994 and 1995. It is the corpus that is used for Dutch in the QA task of CLEF. We selected two question types that are frequent in the CLEF QA question set[3] (Magnini et al, 2003): *capital-of* an *soccer-player-club*. These are binary relations, respectively between a location (e.g. *France*) and its capital (*Paris*) and between a soccer player (e.g. *Dennis Bergkamp*) and his club (*Ajax*). [4]

The *capital-of* relation if functional, i.e. for a given country there is only a single capital. The *soccer-player-club* relation is not one-to-one, as a club has many players, and players can also be playing for more than one club during the two year period covered by the CLEF corpus.

For each of the two question types we created ten seed facts which are listed in Table 1. The initial seeds were chosen with some care since they form the basis of the learning process. For instance, we included not only national capitals, but also the capital of a Dutch province (*Drenthe*) and *Brussels*, the capital of *Europe*. We also included both the adjectival form of a country name as well as the name itself to cover a greater variety of patterns. For the football seeds we made sure the seeds were instances that were true for the period 1994-1995, and we included both full names and last names.

We selected only patterns with a frequency higher than one. In the most lenient experiment all these patterns were used to extract new facts in the CLEF corpus. We performed three experiments: retaining all patterns (i.e. $\tau_p = 0.0$), retaining only patterns with a precision $P \geq 0.5$ ($\tau_p = 0.5$) and retaining only patterns with $P \geq$

[3] http://clef-qa.itc.it/2005/resources.html

[4] Mur (2008) also presents results for learning the ternary *minister-country-department* relation.

Country	Capital	Person	Club
Amerikaanse	Washington	Litmanen	Ajax
Bulgaarse	Sofia	Marc Overmars	Ajax
Drenthe	Assen	Wim Jonk	Inter
Duitsland	Berlijn	Dennis Irwin	Manchester United
Europese	Brussel	Desailly	AC Milan
Frans	Parijs	Romario	Barcelona
Italiaans	Rome	Erwin Koeman	PSV
Bosnisch	Sarajevo	Jean-Pierre Papin	Bayern München
Rusland	Moskou	Roberto Baggio	Juventus
Spaans	Madrid	Aron Winter	Lazio Roma

Table 1 Ten *capital-of* and *soccer-player-club* seeds

Question	Answers
Wat is de hoofdstad van Canada?	Ottawa
Wat is de hoofdstad van Cyprus?	Nicosia
Wat is de hoofdstad van Haïti?	Port-au-Prince
Bij welke club speelt Andreas Brehme?	1. FC Kaiserslautern
Bij welke club speelt Aron Winter?	Lazio Roma; Lazio
Bij welke club speelt Baggio?	Juventus; Milan

Table 2 Sample of question used for evaluation

0.75, where P is computed as described above. This process is repeated for two iterations or until we find more than 5,000 relation instance pairs. Note that we count each matching occurrence of an instance pair in the corpus individually. The reason for this is that for QA it is important to be able to *justify* an answer: i.e. for a given question, not only the answer should be provided, but also the sentence or paragraph from which it was extracted. An answer is *justified* only if this surrounding context provides support for the truth of the answer.

After the extraction stage, we obtain tables that can be used to provide answers for *capital-of* and *soccer player* questions. To test the effect of including these tables in our QA system *Joost*, we expanded the number of relevant questions in the CLEF QA test sets with a number of questions that we have created ourselves. To find more *capital-of* questions we googled for *Wat is de hoofdstad van (What is the capital of)*. Football questions were created by asking five people names of famous football players in 1994 and 1995. For each name in the responses, we created a question of the form *Bij welke club speelde X? (For which club did X play?)*. We checked for all questions that an answer was present in the CLEF corpus. In the end we tested on 42 capital questions and 66 football questions. A few example questions with their answers are given in Table 2.

4.3 Evaluation

We evaluated the extraction relation instances by estimating their precision, and by incorporating them in *Joost* as tables that can be used to provide answers to user questions. We estimated the precision based on a random sample of 100 relation instance pairs. To evaluate the results of the QA system, we simply counted the number of times the first answer was correct, and we computed the mean reciprocal rank over the first 5 answers. The reciprocal rank score for a question is $1/R$, where R is the rank of the first correct answer.

The results are given in Tables 3 and 4. Using no filtering (i.e. $P \geq 0.0$), we found 39 patterns for *capital-of* in the first round (i.e. using only the seeds given in Table 1). Applying these 39 patterns we extracted 3,405 new instance pairs. The estimated precision is 0.58. When using these instance pairs as a table for off-line question answering in *Joost*, we answered 35 of 42 questions correctly. The mean reciprocal rank was 0.87. For the second round we repeated the process using 3,405 instance pairs. With these facts we found 1,306 patterns. The 1,306 patterns in turn returned 234,684 instance pairs.

The middle and bottom part of the tables show the results for the experiment in which we filter the patterns using $P \geq 0.5$ and $P \geq 0.75$. for the *capital-of* relation we stopped after two iterations, for the *soccer-player-club* relation we stopped after we found more than 5,000 facts. The best performance per category is marked in bold.

Capital-of $P \geq 0.0$	# patterns	# pairs	(P)	1st ans OK (# q = 42)	MRR
1st round	39	3,405	(0.58)	35	0.87
2nd round	1306	234,684	(0.01)	14	0.51
$P \geq 0.5$					
1st round	24	2,875	(0.63)	35	0.85
2nd round	171	4,123	(0.49)	**37**	**0.90**
$P \geq 0.75$					
1st round	17	2,076	(0.83)	35	0.84
2nd round	64	2,344	(0.83)	35	0.85

Table 3 Results for learning patterns for answering *capital-of* questions.

The results for using automatically created relation instance tables in the QA system show that even low precision data can help to improve the performance of the QA system. The best results for answering *capital-of* questions were obtained in the second round with $P \geq 0.50$. The precision of the extracted facts was only a mediocre 0.49 compared to 0.83 in the experiments with $P \geq 0.75$. The number

Football $P \geq 0.0$	# patterns	# pairs	(P)	1st ans OK (# q = 66)	MRR
1st round	19	115,296	(0.01)	40	0.66
$P \geq 0.5$					
1st round	11	109,435	(0.01)	**41**	**0.67**
$P \geq 0.75$					
1st round	6	196	(0.26)	11	0.17
2nd round	28	31,958	(0.02)	18	0.31

Table 4 Results for learning football patterns (Best results in bold)

of facts, on the other hand, was almost twice as high (4,123 vs. 2,344). The QA evaluation shows that 'recall' is more important that precision in this case.

Table 4 illustrates this effect even more strongly. The best result is again $P \geq 0.50$, but this time there is no significant difference with the result for experiment where no pattern filtering was applied. An extremely large number of incorrect instance pairs was extracted in both cases, but this did not hurt performance on the QA task.

This rather contradictory result can be explained by the patterns that were found for the extraction of football facts. A very frequent pattern we found was *Player* $\rightarrow$ *mod+Club*. The pattern occurs typically when a name is followed by a name in brackets, i.e. it matches for example with the phrase *Jari Litmanen (Ajax)* but also with *Rijksuniversiteit Groningen (RUG)*. Although the pattern is noisy, the incorrect facts typically have nothing to do with football players, and thus they do not cause incorrect answers to football questions (where the name of the player is always given).

The results of the experiments for the extraction of *capital-of* and *soccer player-club* instance pairs suggest that for the benefit of off-line QA it is better to focus on high recall than on high precision. The use of a pattern filtering method based on the estimated precision of patterns provides a method for balancing precision and recall of the extraction process. It should be noted, however, that this is a rather crude method. The experiments illustrate that varying the value of P used for filtering can easily lead to a situation where either very few instance pairs are extracted or where an excessive number of instances is extracted, with very low precision. The latter situation makes further iterations of the extaction process fruitless (as the level of noise is simply to high). A second problem for the method above is the fact that all relation instances found in iteration I are used as seeds for iteration $I+1$. Given the low precision of some the experiments, it becomes interesting to search for methods that use as seeds only the most reliable instances from a previous round.

The *Espresso* algorithm of Pantel and Pennacchiotti (2006) is a lightly supervised information extraction method that offers a better balance between precision and recall. By selecting only the most reliable patterns and only the most reliable

relation instances in each round of the bootstrapping process, more iterations can be carried out without large drops in precision. The reliability of instances and patterns is computed by means of a scoring criterion based on the mutual information score between instance pairs and patterns. Recent work by Ittoo and Bouma (2010) and Bouma and Nerbonne (2010) uses *Espresso* for relation extraction on a 700M word corpus, containing among others the CLEF corpus. Whereas Pantel and Pennacchiotti (2006) use surface strings as extraction patterns, the experiments described here use dependency patterns for extraction.

Using the seed list for the *soccer-player-club* relation given in Table 1, and using the *Espresso* algorithm for relation extraction, we obtain the results in Table 5. The first two columns give results for using the method of Pantel and Pennacchiotti (2006). As in that paper, initially the 10 highest scoring patterns are selected, and 100 instance pairs and 1 pattern per iteration are added. Precision in all iterations is as good or better as that of the experiment with the highest precision in Table 4. As with the experiments above, however, iterative, lightly supervised methods like this are subject to *semantic drift*, i.e. the phenomenon that errors in previous iterations have a deteriorating effect on the accuracy of later iterations McIntosh and Curran (2009). To dampen this effect, distributional similarity (Lin, 1998; van der Plas, 2008) was used to filter instance pairs where the first element is not distributionally similar to the group of *soccer players* or where the second element in not similar to *soccer clubs*. The results for this method are given in the final two columns.

	Espresso		Espresso$^+$	
	pairs	prec	pairs	prec
1st round	109	0.36	40	0.65
2nd round	211	0.30	74	0.53
3rd round	312	0.25	88	0.38
4th round	412	0.27	176	0.41
5th round	511	0.33	290	0.45

Table 5 Accuracy per iteration for learning the *soccer* relation using *Espresso* and *Espresso* combined with a distributional similarity filter.

It is hard to compare the recall of the technique we used for creating tables for QA and the relation extraction method based on *Espresso*. Most importantly, for QA we counted individual occurrences of instance pairs (as we need the context of each instance pair as *justification* of the answer) whereas for general purpose relation extraction all occurrences of an instance pair are counted as a single instance. If we count individual occurrences, the number of instances retrieved for the *Espresso* experiments is almost 12,000, whereas for the system with filtering it is over 7,000. This suggests that recall might still be sufficient for integration of the *Espresso* technique in a QA system, but we have not tested this at the time of writing.

5 Relation extraction for medical QA

Relation extraction for open domain QA has concentrated on relation types where the arguments are named entities and dates. In the large corpora used for open domain QA, these occur relatively frequent, with little variation in spelling. Thus, lightly supervised methods, that rely on the fact that many instantiations of a relation are present in the corpus, and that some of these are highly frequent, has been used succesfully as a component in open domain QA. For closed-domain, medical, QA, the situation is more complex. Here, the relations of interest typically exist between concepts expressed as complex noun phrases. An example is given in (8).

(8) De ziekte van Graves-Basedow (toxische diffuse struma) wordt vermoedelijk veroorzaakt door een antilichaam dat de schildklier aanzet tot overproductie van het schildklierhormoon. (*Graves' disease (toxic diffuse goitre) is most likely caused by an anti body which leads the the thyroid to excessive production of thyroid hormone.*)

This sentence expresses a causal relation between a disease (*De ziekte van Graves-Basedow* and a cause, *een antilichaam dat de schildklier aanzet tot overproductie van het schildklierhormoon*. It is unlikely that these exact two phrases will ever occur frequently as a pair in the corpus. As this is true for most of the instances of the *cause* relation in the corpus, the chances of bootstrapping extraction patterns from seeds for the cause relation are not very promising. Most work on medical relation extraction has therefore used at least some amount of annotated data. In this section, we describe how relation instances can be extracted from a Dutch medical corpus, using annotated data and UMLS to guide the extraction process.

Within the IMIX project, a substantial corpus of Dutch medical text has been annotated with semantic labels.[5] The annotated corpus (approx. 600K words) consists of texts from a medical encyclopedia and a medical handbook. An example of the annotation is given in Figure 2.

```
<rel_causes>
   Een tekort aan
          <con_body_part>insuline</con_body_part>
   leidt tot
          <con_disease>suikerziekte</con_disease>
</rel_causes>
```

Fig. 2 Semantic annotation in the IMIX corpus (of the sentence *A shortage of insuline leads to diabetes*).

Sentences may be labeled with relation tags, such as *rel_causes*. Noun phrases denoting concepts are annotated with one of 12 semantic concept types such as

[5] developed by the University of Tilburg IMIX/Rolaquad project (ilk.uvt.nl/rolaquad).

body_part or *disease*. Seven different relation types are present in the corpus: *causes, has_definition, diagnoses, occurs, prevents, has_symptom* and *treats*). The frequency of each relation ranges from 479 (*prevents*) to 4,301 (*treats*). Some sentences are annotated with more than one relation type. This annotation is very helpful for learning medical relation extraction patterns, although it should be noted that no labeling is present which explicitly identifies the arguments of a relation, and furthermore, that it is not guaranteed that suitable arguments for a relation can actually be found within the sentence. For instance, in (9), the subject *dit alles* (*all this*) is an anaphoric expression which is not in itself a suitable argument for a medical relation instance pair.

(9) Dit alles duidt op een verschil in ontwikkeling van de hersenen bij jongetjes en meisjes (*All this indicates a difference in development of the brain in boys and girls*)

We used the corpus to learn dependency patterns that are associated with a given medical relation. It turns out to be the case that many relations can be expressed in text by general linguistic patterns (*X may lead to Y, X occurs in Y*), which are not unique to a given medical relation, and also, which do not imply that both *X* and *Y* are medical concepts. Such patterns can nevertheless be used to extract medical relations with high accuracy if we require that both *X* and *Y* are medical terms. We may also impose the restriction that *X* and *Y* have to be terms that belong to a given class (i.e. *X* and *Y* are medical terms denoting, respectively, a *virus* and a *disease*). By imposing restrictions on the semantic class of the argument, we also reduce the ambiguity of the dependency patterns.

Below, we first present a method for predicting concept labels of Dutch medical terms, using (English) UMLS as thesaurus. Next, we present our method for learning extraction patterns based on the IMIX corpus. We then show that the combination of extraction patterns and semantic concept labels provides accurate results for relation extraction. Finally, we evaluate the effect of incorporating the extracted relation instances in a medical QA system.

5.1 Multilingual term labeling

Medical terminology differs across languages, but also is closely related. Technical medical terms in Dutch, for instance, often are simply borrowed from English (i.e. *stress, borderliner, drugs* and acronyms like ADHD and PTSS), or are cognates (i.e. English *genetic* and Dutch *genetisch*). Some terms are genuinely different in the two languages (*infection* and *besmetting*), and need to be translated.

For classifying Dutch terms on the basis of a subset of UMLS concepts that contains Dutch and English terms,[6] we use a sequence of five heuristics, illustrated

[6] The relevant subset of UMLS consists of 163,032 concepts in Dutch and 2,974,889 concepts in English.

in Table 6. If step 1 returns no result, step 2 will be evaluated, and so on. For the example *psychische aandoeningen (mental disorders)*, step 3 returns the result *B2.2.1.2.1.1:Mental or Behavioral Dysfunction*. In case there is more than one result (this happens especially in steps 4 and 5), a further heuristic is needed to decide the best labels. Given a query term QT, we may find a match with a term in UMLS UT, which has concept type UC. If a query returns more than three results, we first restrict results to cases where QT and UT have the same length. If more than three results remain, we filter all Dutch terms UT which have the head in a different position than QT. Finally, the remaining semantic types UC are ranked by frequency, and the three highest ranked types are selected.

No	Lang	Index	Query parameters	Example
1	NL	root	exact match of root forms	psychish aandoening
2	NL	term	exact match of term string	psychische aandoeningen
3	EN	term	exact match of translated term[7]	mental disorders
4	DU	head	exact match of head word	aandoening
5	NL or EN	term	one of the words in the term	psychisch OR mental OR ...

Table 6 Five conditional steps in classifying a Dutch term, *psychische aandoeningen* (*'mental disorders'*), based on a subset of UMLS concepts in Dutch (NL) and English (EN).

The contribution of each of the heuristics was evaluated by applying the method pages in the category Health Care of Dutch Wikipedia. 370,578 terms were found. 17% of these were found in Dutch part of the UMLS and 30% in English part (through translation); 38% are new terms which could be assigned a concept label in steps 4 and 5, and 16% of the terms received no label. For the new terms, 26% were labeled using heuristic 4 (matching Dutch head word) and 74% using heuristic 5 (a matching Dutch or English word). This shows that labeling new Dutch terms benefits from reusing labels of existing terms, and from translation.

We evaluated the accuracy of our method on the 1000 most frequent terms in the IMIX medical corpus. As our system returns (one of 135) UMLS semantic concepts, whereas the IMIX corpus uses (only 12) corpus-specific concept labels, we defined a (many to one) mapping from UMLS labels to more corpus concept labels. Note that each UMLS label was mapped to at most one corpus label. Evaluation is done on the basis of the highest ranked UMLS concept assigned by the heuristics outlined above. We obtained a precision of 78.2%.

It should be noted that the mapping creates certain mismatches. For instance, the term *tandsteen* (*calculus*) was labeled as *disease* in the corpus but as *Body Substance* in UMLS (and by our classification method). We believe both classes are correct, although in general it is not correct to map *Body Substance* to *disease*. Thus, we suspect that the actual accuracy of our method may be slightly higher than the precision figure suggests.

Canisius et al (2006) use a machine learning approach to train a concept classifier for the same data. They do not use external resources, but instead try to learn the

classification from (a subset of) the corpus itself. They report an accuracy of 68.9% and a theoretical upper bound of 74.9%. This suggests that, given the limited size of the corpus, the use of external knowledge sources (UMLS in particular) boosts the performance of concept labeling.

5.2 Learning patterns

Given two medical terms in a sentence labeled with a medical relation, we extract the shortest dependency path connecting the two as an extraction pattern. For instance, given sentence (10-a), we may extract among others the patterns (10-b) and (10-c).

(10) a. Aantasting van de bijnierschors door infecties (bijv. tuberculose) of bij auto-immuunziekten kan leiden tot de ziekte van Addison (*Erosion of the adrenal glands by infections (eg, tuberculosis) or with autoimmune diseases could lead to Addison's disease*)
b. ARG1+*obj1*+*van*+*mod*+*aantasting*+*subj* ← *leid* → *pc*+*tot*+ARG2
c. ARG1+*subj* ← *leid* → *pc*+*tot*+ARG2

We require that both ARG1 and ARG2 must match a medical term. In (10-a) the case for *aantasting van de bijnierschors* and *ziekte van Addison*, and thus one of the patterns we obtain is:

(11) NEOPLASTICPROCESS+*subj* ← *leid* → *pc*+*tot*+DISEASEORSYNDROME

(11) is an example of a *semantic extraction pattern*, i.e. a dependency pattern with semantic (UMLS) class labels for ARG1 and ARG2. To find the appriate semantic label for a complex argument, we first extract its main term using a linguistic filter adapted from (Justeson and Katz, 1995). The filter extracts a sub-string of the argument that matches the following POS-tag regular expression:

(12) `((Adj | N)* N Prep Det?)? (Adj | N)* N`

For the subject in the example above, it extracts *Aantasting van de bijnierschors* (`N Prep Det N`) as the main term. Next, we find semantic class labels for the term using the method outlined in section 5.1.

The task of pattern learning is to find sets of semantic extraction patterns for each of the relations in the IMIX corpus. For each pair of medical concept terms in a sentence, we extract the dependency path connecting the two. For the main terms in the concepts, we determine the UMLS class labels. Where this returns more than one concept label, all combinations of concept labels are used to generate semantic extraction patterns. For instance, for (10-a) we obtain the following semantic relation patterns:

(13) a. DISEASEORSYNDROME+*subj* ← *leid* → *pc*+*tot*+DISEASEORSYNDROME
b. NEOPLASTICPROCESS+*subj* ← *leid* → *pc*+*tot*+DISEASEORSYNDROME

c. FINDING+*subj* ← *leid* → *pc*+*tot*+DISEASEORSYNDROME

Dependency patterns are ranked according to the relative frequency with which they occur with a given relation and patterns below a certain treshold are discarded. For dependency pattern ranking, we compute the weight of a pattern R as the ratio of the probability $P(R_C)$ with which R was found in a training corpus C containing only sentences labeled with the relevant relation, and its probability $P(R_G)$ in the general medical corpus G. We multiply this score with the frequency of R in C, as shown below:

$$weight(R) = \frac{P(R)}{P(R_G)} \times f(R_C)$$

The intuition behind this method is that good patterns ought to appear more frequently in the training corpus for the relation than in the general corpus. We multiple with frequency again to decrease the importance of low frequency patterns in the training corpus.

Semantic relation patterns consist of dependency patterns extended with semantic labels. Semantic relation patterns $R_{(A,B)}$ are weighted by multiplying their frequency with the weight of their *dependency patterns R*:

$$weight(R_{(A,B)}) = weight(R) \times f(R_{(A,B)})$$

During relation extraction for a given relation *Rel*, we extract from a corpus all relation instances *R(*ARG1,ARG2*)*, where the dependency pattern *R* has to be a valid dependency pattern for *Rel* and the semantic types of `Arg1` and `Arg2` have to match one of the semantic relation patterns $R_{(A,B)}$ for *Rel*.

To investigate the effect of concept labeling, we also carried out relation extraction experiments where both arguments of a potential relation instance did not match the semantic types of a semantic relation pattern, but instead only one or no argument matched.

5.3 Evaluation

We evaluated the accuracy of the semantic relation extraction patterns on a subset of the IMIX corpus, and on text from the Health Care section of Wikipedia.

From the IMIX corpus, we randomly selected for each relation 50 sentences as test set. Note that these were withheld from the corpus that was used for learning the extraction patterns. We selected from these 50 sentences only those that contain two fully specified arguments of the relation and discarded from the test set sentences containing e.g. anaphoric NPs as argument. Note that, since relation labeling was done at the level of sentences, for many relations, less than 50% of the labeled instances actually contain both arguments of the relation. This indicates that the

corpus is a good deal less informative than corpora which explicitly mark relations between (medical) terms. Table 7 gives the results for the various relations.

Relation type	# pat.	P	R	F
has_definition	4	0.83	0.73	0.78
causes	155	0.92	0.67	0.77
occurs	71	0.81	0.54	0.65
has_symptom	206	0.58	0.62	0.60
prevents	47	0.80	0.40	0.53
treats	180	0.71	0.40	0.51
diagnoses	85	0.86	0.24	0.38

Table 7 The relation extraction results on the test data of 50 sentences per relation type. #pat is the number of semantic extraction patterns for that relation.

Precision (i.e. the number of times relation label *R* was correctly predicted divided by the number of times this label was predicted by the system) is relatively high for all patterns, but recall varies. The method performs reasonably well for the `has_definition` (f-measure .78) and `causes` (.77) relation types, and performs less well for the `diagnoses` relation (.38). Variation in performance is probably due to the fact that for some relations, more training examples are available, some relations are expressed by simpler dependency patterns (i.e. *is caused by*), and some patterns seem to suffer more from parsing errors.

The relations `causes` and `has_symptom` have many similar depedency patterns (*be_cause_by*). However, the semantic classes of their arguments are different. In the first ten relation patterns for `causes` the object argument has diverse semantic types: *Disease or Syndrome* (3), *Finding* (2), *Pathologic Function*, *Functional Concept*, *Protein*, *Bacterium*, and *Virus*, while `has_symptom` is dominated by *Sign or Symptom* (6). This shows that the semantic type of a term, along with its pattern, plays an important role in identifying the type of a relation instance.

To further evaluate the effect of using semantic types in the relation extraction task, we tested our patterns on a subset of Wikipedia text that contains medical lemmas. We randomly selected 20 extraction results for each relation and each level of matching of the semantic argument types, and evaluated these manually. The number of results for each level and the precision is given in Table 8.

For all of the relation types, the best accuracy is at level 2M, where both of the arguments have semantic types matching a pattern for that relation. The drop in precision at level 0M (where no matching argument was found) is considerable. For the `prevents` relation type, 50% of the errors at 1M are cases where one of the arguments is a definite NP. To obtain a full interpretation of these NPs, they need to be interpreted as coreferential with a preceding NP. Although the generalization of patterns increases recall, it also becomes the main cause of errors at 1M and, especially, 0M. The noise in the training data also contributes to errors, although the pattern filtering has reduced a great amount of irrelevant patterns. Another source of

Relation type	2M		1M		0M	
	#	prec	#	prec	#	prec
causes	942	0.95	1,625	0.90	647	0.75
has_definition	4,102	0.95	6,118	0.65	657	0.30
occurs	548	0.90	2,219	0.80	1,238	0.50
treats	300	0.85	1,826	0.60	1,026	0.45
has_symptom	1,220	0.80	2,668	0.30	850	0.00
prevents	24	0.75	171	0.50	470	0.50
diagnoses	34	0.60	265	0.60	231	0.35
total	7,170	0.83	14,892	0.62	5,149	0.41

Table 8 Number of matching dependency relation patterns per relation with 2, 1 , or 0 matching concept labels.

errors are non-medical term arguments, such as name of places, concepts for other domains, or non-term arguments.

The low accuracy scores at 0M are a further indication that the coverage of our concept labeling system is satisfactory: if both of the arguments cannot be assigned a medical concept label, we can be relatively certain that the extracted arguments are not proper instances of the relation.

5.4 Evaluation in a QA *setting*

In this section, we report on the performance of the QA system *Joost* on a test suite of questions from the medical domain. The test suite was derived from the pool of 435 candidate questions by Tjong Kim Sang et al (2005), and expanded with questions found on the web (by submitting keywords and phrases from typical medical questions to a search engine). Many of the candidate questions found on the web have no answer in the IMIX medical corpus. We selected only questions which have at least one answer in the corpus.

We tested the performance of the QA system on 58 questions, covering three question types: *has_definition* (25 questions), *causes* (22 questions) and *has_symptom* (11 questions). The results of the performance of the QA system in the three experimental settings are shown in Table 9. Here, *manual* refers to the QA system using tables created using manually constructed extraction patterns, *learned* refers to the system using tables based on automatically learned extraction patterns (as described above), and IR refers to the QA system without any tables but relying solely on passage retrieval and answer extraction. Results are given in terms of mean reciprocal rank MRR (i.e. the mean of $1/R$, where R is the rank of first correct answer) and *1st* correct.

In general, the MRR and first answer correct scores of the manual and the pattern learning methods outperform the scores of the IR baseline. For all of the question types, the performance scores of the system using automatically learned extraction patterns outperform the scores of the manual method. And overall, our method con-

method	answered	MRR	1st
has_definition			
manual	16	0.333	0.280
learned	**22**	**0.465**	**0.360**
IR	21	0.133	0.040
causes			
manual	19	0.547	0.409
learned	**20**	**0.750**	**0.682**
IR	19	0.405	0.318
has_symptom			
manual	5	0.364	0.364
learned	8	**0.636**	**0.636**
IR	8	0.182	0.182
overall			
manual	40	0.420	0.345
learned	**50**	**0.605**	**0.534**
IR	48	0.246	0.172

Table 9 Performance scores of the question answering system on the three experiment settings (manual, learned, and IR), measured using MRR and first answer correct, for the three question types (*has_definition*, *causes*, and *has_symptom*).

tributes 42% and 52% improvement against the manual method with respect to the MRR score and first answer correct, respectively. This shows that our method has successfully improved the performance of the medical QA system.

6 Conclusions and future work

In this chapter, we have stressed the importance of relation extraction for boosting the performance of QA systems. This is true for both open-domain QA and specialized QA such as medical QA. Our experiments in open-domain QA have concentrated on methods that have high recall, sometimes at the expense of precision. In the context of a QA system, much of the noise incorporated by high coverage extraction patterns never surface, as user questions always supply one argument of the relation, and also, because the frequency with which pairs are found is used to rank answers. Correct answers tend to be extracted more often than incorrect ones, even in systems that introduce substantial levels of noise.

The experiments on medical relation extraction cannot rely on the fact that (seed) instance pairs are frequent in the corpus, and that arguments to a given relation are easily found. To overcome the problem of identifying terms denoting medical terms, we presented a method for assigning UMLS concept labels to Dutch medical terms which employs both the Dutch and English part of UMLS. We showed that both the coverage and the precision of our term classification method is relatively high

compared with other methods that do not use external knowledge. This experiment also shows that it is possible to use a multilingual resource to classify new terms in a particular language.

Relation patterns for medical relation extraction can be obtained from sentences that were labeled with only the relation they contain. Concept labeling helps in improving the accuracy of relation extraction: it is used to rank relevant patterns higher, to distinguish identical dependency patterns for different relations, and to predict which matching patterns in a test corpus are most likely correct instances of the relation.

Our current method uses the semantic types and semantic relation (patterns) from the training data. In the future we plan to (semi-automatically) annotate corpora using the UMLS Semantic Network that contains 135 semantic types and 54 relationships.

The relative success of using dependency patterns with concept labels in the medical domain suggests that similar methods might also provide a means to improve the accuracy of open-domain relation extraction. In particular, term identification might help to detect term variation, also for person, organization, and geographical names, and could help to find multiword terms. Concept labeling could help to reduce the level of noise in our current open-domain relation extraction system.

During the evaluation of the medical relation extraction results, we noticed that an important source of errors was due to coreference. Sentences such as *This form is transferred via a dominant gene*, or *The disease is caused by a surplus of growth hormone* are labeled as *cause*, but were discarded as *cause* sentences from the gold standard evaluation set, as they do not contain complete information for one of the arguments of the relation. We estimated that approximately 9% of the relation candidates in the Wikipedia data contains a pronominal or definite NP that needs anaphoric interpretation. An obvious next step would be to apply coreference resolution to medical terms, so as to obtain a full interpretation of the term, and a term which can be used for concept classification.

References

Bodenreider O (2004) The unified medical language system (UMLS): integrating biomedical terminology. Nucleic Acids Research 32(Database Issue):D267

Bouma G, Nerbonne J (2010) Applying the espresso-algorithm to large parsed corpora

Bouma G, van Noord G, Malouf R (2001) Alpino: Wide-coverage computational analysis of Dutch. In: Computational Linguistics in The Netherlands 2000, Rodopi, Amsterdam

Bouma G, Fahmi I, Mur J, van Noord G, van der Plas L, Tiedeman J (2005) Linguistic knowledge and question answering. Traitement Automatique des Langues 2(46):15–39

Bouma G, Mur J, van Noord G, van der Plas L, Tiedemann J (2006) Question answering for dutch using dependency relations. In: Peters C (ed) Accessing Multilingual Information Repositories, pp 370–379, URL http://dx.doi.org/10.1007/11878773_42

Braun L, Wiesman F, van den Herik J (2005) Towards automatic formulation of a physician's information needs. In: Proceedings of the Dutch-Belgian Information Retrieval Workshop, Utrecht, the Netherlands

Briscoe T, Carroll J (2002) Robust accurate statistical annotation of general text. In: Proceedings of the 3rd International Conference on Language Resources and Evaluation, Citeseer, pp 1499–1504

Bunescu R, Mooney R (2005) A shortest path dependency kernel for relation extraction. In: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, Vancouver, pp 724–731

Canisius S, van den Bosch A, Daelemans W (2006) Constraint satisfaction inference: Non-probabilistic global inference for sequence labelling. In: Proceedings of the EACL 2006 Workshop on Learning Structured Information in Natural Language Applications, Trento

Culotta A, Sorensen J (2004) Dependency tree kernels for relation extraction. In: 42nd Annual Meeting of the Association for Computational Linguistics (ACL), Barcelona, Spain

Etzioni O, Cafarella M, Downey D, Popescu A, Shaked T, Soderland S, Weld D, Yates A (2005) Unsupervised named-entity extraction from the web: An experimental study. Artificial Intelligence 165(1):91–134

Fleischman M, Hovy E, Echihabi A (2003) Offline strategies for online question answering: Answering questions before they are asked. In: Proc. 41st Annual Meeting of the Association for Computational Linguistics, Sapporo, Japan, pp 1–7

Fundel K, Küffner R, Zimmer R (2007) Relex - relation extraction using dependency trees. Bioinformatics 23:365–371

H

”am

”al

”ainen A, Boves L, de Veth J (2005) Syllable-length acoustic units in large-vocabulary continuous speech recognition. In: Proceedings of SPECOM, Citeseer, pp 499–502

Han Y, Veth J, Boves L (2005) Speech Trajectory Clustering for Improved Speech Recognition. In: Ninth European Conference on Speech Communication and Technology, Citeseer

Ittoo A, Bouma G (2010) Mereological and meronymic relations for learning part whole relations. In: Computational Linguistics in the Netherlands 2010, Utrecht, the Netherlands

Jijkoun V, Mur J, de Rijke M (2004) Information extraction for question answering: Improving recall through syntactic patterns. In: Coling 2004, Geneva, pp 1284–1290

Justeson J, Katz S (1995) Technical terminology: some linguistic properties and an algorithm for identification in text. Natural language engineering 1(01):9–27

Katrenko S, Adriaans P (2007) Learning relations from biomedical corpora using dependency trees. In: Tuyls K, Westra R, Saeys Y, Nowé A (eds) Knowledge Discovery and Emergent Complexity in BioInformatics, Lecture Notes in Bioinformatics. LNBI, vol. 4366, Springer

Lin D (1998) Automatic retrieval and clustering of similar words. In: Proceedings of COLING/ACL, Montreal, pp 768–774

Lin D (2003) Dependency-based evaluation of MINIPAR. Treebanks: building and using parsed corpora p 317

Lin D, Pantel P (2001) Discovery of inference rules for question answering. Natural Language Engineering 7:343–360

Lita L, Carbonell J (2004) Unsupervised question answering data acquisition from local corpora. In: Proceedings of the thirteenth ACM international conference on Information and knowledge management, ACM, p 614

Magnini B, Romagnoli S, Vallin A, Herrera J, Peñas A, Peinado V, Verdejo F, de Rijke M (2003) The multiple language question answering track at clef 2003. In: Peters C (ed) Working Notes for the CLEF 2003 Workshop, Trondheim, Norway

McCarthy D, Koeling R, Weeds J, Carroll J (2007) Unsupervised acquisition of predominant word senses. Computational Linguistics 33(4):553–590

McIntosh T, Curran J (2009) Reducing semantic drift with bagging and distributional similarity. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP

Mur J (2008) Off-line answer extraction for question answering. PhD thesis, University of Groningen, Groningen

van Noord G (2004) Error mining for wide-coverage grammar engineering. In: Proceedings of the ACL 2004, Barcelona

van Noord G (2006) At last parsing is now operational. In: Mertens P, Fairon C, Dister A, Watrin P (eds) TALN06. Verbum Ex Machina. Actes de la 13e conference sur le traitement automatique des langues naturelles, pp 20–42

van Noord G (2009) Learning efficient parsing. In: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, Athens, Greece, pp 817–825

Padó S, Lapata M (2007) Dependency-based construction of semantic space models. Computational Linguistics 33(2):161–199

Pantel P, Pennacchiotti M (2006) Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In: Proceedings of Conference on Computational Linguistics / Association for Computational Linguistics (COLING/ACL-06), Sydney, Australia, pp 113–120

van der Plas L (2008) Automatic lexico-semantic acquisition for question answering. PhD thesis, University of Groningen

Pollard C, Sag I (1994) Head-driven Phrase Structure Grammar. Center for the Study of Language and Information Stanford

Prins R, van Noord G (2001) Unsupervised pos-tagging improves parsing accuracy and parsing efficiency. In: IWPT 2001: International Workshop on Parsing Technologies, Beijing China

Ravichandran D, Hovy E (2002) Learning surface text patterns for a question answering system. In: Proceedings of ACL, vol 2, pp 41–47

Rinaldi F, Schneider G, Kaljurand K, Hess M, Romacker M (2006) An environment for relation mining over richly annotated corpora: the case of genia. BMC Bioinformatics 7

Rosario B, Hearst M (2004) Classifying semantic relations in bioscience texts. In: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, Barcelona, Spain

Snow R, Jurafsky D, Ng A (2005) Learning syntactic patterns for automatic hypernym discovery. Advances in Neural Information Processing Systems 17:1297–1304

Soubbotin M, Soubbotin S (2002) Use of patterns for detection of answer strings: A systematic approach. In: Proceedings of TREC, vol 11

Stevenson M, Greenwood M (2009) Dependency pattern models for information extraction. Research on Language and Computation 3:13–39

Tiedemann J (2005) Integrating linguistic knowledge in passage retrieval for question answering. In: Proceedings of EMNLP 2005, Vancouver, pp 939–946

Tjong Kim Sang E, Bouma G, de Rijke M (2005) Developing offline strategies for answering medical questions. In: Mollá D, Vicedo JL (eds) AAAI 2005 workshop on Question Answering in Restricted Domains

Zhao S, Grishman R (2005) Extracting relations with integrated information using kernel methods. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, Ann Arbor, Michigan, pp 419 – 426