Named Entity Recognition for Question Answering

Diego Mollá and **Menno van Zaanen** and **Daniel Smith**
Centre for Language Technology
Macquarie University
Sydney
Australia
{diego, menno, dsmith}@ics.mq.edu.au

Abstract

Current text-based question answering (QA) systems usually contain a named entity recogniser (NER) as a core component. Named entity recognition has traditionally been developed as a component for information extraction systems, and current techniques are focused on this end use. However, no formal assessment has been done on the characteristics of a NER within the task of question answering. In this paper we present a NER that aims at higher recall by allowing multiple entity labels to strings. The NER is embedded in a question answering system and the overall QA system performance is compared to that of one with a traditional variation of the NER that only allows single entity labels. It is shown that the added noise produced introduced by the additional labels is offset by the higher recall gained, therefore enabling the QA system to have a better chance to find the answer.

1 Introduction

Many natural language processing applications require finding named entities (NEs) in textual documents. NEs can be, for example, person or company names, dates and times, and distances. The task of identifying these in a text is called named entity recognition and is performed by a named entity recogniser (NER).

Named entity recognition is a task generally associated with the area of information extraction (IE). Firstly defined as a separate task in the Message Understanding Conferences (Sundheim, 1995), it is currently being used in a varied range of applications beyond the generic task of information extraction, such as in bioinformatics, the identification of entities in molecular biology (Humphreys et al., 2000), and text classification (Armour et al., 2005).

In this paper we will focus on the use of named entity recognition for question answering. For the purposes of this paper, question answering (QA) is the task of automatically finding the answer to a question phrased in English by searching through a collection of text documents. There has been an increase of research in QA since the creation of the question answering track of TREC (Voorhees, 1999), and nowadays we are starting to see the introduction of question-answering techniques in mainstream web search engines such as Google[1], Yahoo![2] and MSN[3].

An important component of a QA system is the named entity recogniser and virtually every QA system incorporates one. The rationale of incorporating a NER as a module in a QA system is that many fact-based answers to questions are entities that can be detected by a NER. Therefore, by incorporating in the QA system a NER, the task of finding some of the answers is simplified considerably.

The positive impact of NE recognition in QA is widely acknowledged and there are studies that confirm it (Noguera et al., 2005). In fact, virtually every working QA system incorporates a NER. However, there is no formal study of the optimal characteristics of the NER within the context of QA. The NER used in a QA system is typically developed as a stand-alone system designed independently of the QA task. Sometimes

[1] http://www.google.com
[2] http://search.yahoo.com
[3] http://search.msn.com

Proceedings of the 2006 Australasian Language Technology Workshop (ALTW2006), pages 51–58.

it is even used as a black box that is not fine-tuned to the task. In this paper we perform a step towards such a formal study of the ideal characteristics of a NER for the task of QA. In particular, section 2 comments on the desiderata of a NER for QA. Next, section 3 describes the QA system used in the paper, while section 4 describes the NER and its modifications for its use for QA. Section 5 presents the results of various experiments evaluating variations of the NER, and finally Section 6 presents the concluding remarks and lines of further research.

2 Named Entity Recognition for Question Answering

Most QA systems gradually reduce the amount of data they need to consider in several phases. For example, when the system receives a user question, it first selects a set of relevant documents, and then filters out irrelevant pieces of text of these documents gradually until the answer is found.

The NER is typically used as an aid to filter out strings that do not contain the answer. Thus, after a question analysis stage the type of the expected answer is determined and mapped to a list of entity types. The NER is therefore used to single out the entity types appearing in a text fragment. If a piece of text does not have any entity with a type compatible with the type of the expected answer, the text is discarded or heavily penalised. With this in mind, the desiderata of a NER are related with the range of entities to detect and with the recall of the system.

2.1 Range of Entities

Different domains require different types of answers. Typically, the question classification component determines the type of question and the type of the expected answer. For example, the questions used in the QA track of past TREC conferences can be classified following the taxonomy shown in Table 1 (Li and Roth, 2002).

The set of entity types recognised by a standalone NER is typically very different and much more coarse-grained. For example, a typical set of entity types recognised by a NER is the one defined in past MUC tasks and presented in Table 2. The table shows a two-level hierarchy and the types are much more coarse-grained than that of Table 1. Within each of the entity types of Table 2 there are several types of questions of Table 1.

ABBREVIATION abb, exp
ENTITY animal, body, color, creative, currency, dis.med., event, food, instrument, lang, letter, other, plant, product, religion, sport, substance, symbol, technique, term, vehicle, word
DESCRIPTION definition, description, manner, reason
HUMAN group, ind, title, description
LOCATION city, country, mountain, other, state
NUMERIC code, count, date, distance, money, order, other, period, percent, speed, temp, size, weight

Table 1: Complete taxonomy of Li & Roth

Class	**Type**
ENAMEX	Organization
	Person
	Location
TIMEX	Date
	Time
NUMEX	Money
	Percent

Table 2: Entities used in the MUC tasks

A QA system typically uses both a taxonomy of expected answers and the taxonomy of named entities produced by its NER to identify which named entities are relevant to a question. The question is assigned a type from a taxonomy such as defined in Table 1. This type is then used to filter out irrelevant named entities that have types as defined in Table 2.

A problem that arises here is that the granularity of the NEs provided by a NER is much coarser than the ideal granularity for QA, as the named entity types are matched against the types the question requires. Consequently, even though a question classifier could determine a very specific type of answer, this type needs to be mapped to the types provided by the NER.

2.2 Recall

Given that the NER is used to filter out candidate answers, it is important that only wrong answers are removed, while all correct answers stay in the set of possible answers. Therefore, recall in a NER in question answering is to be preferred above precision. Generally, a NER developed for a generic NE recognition task (or for information extraction) is fine-tuned for a good balance between recall and precision, and this is not necessarily what we need in this context.

2.2.1 Multi-labelling

Recognising named entities is not a trivial task. Most notably, there can be ambiguities in the detection of entities. For example, it can well happen that a text has two or more interpretations. Notable examples are names of people whose surname takes the form of a geographical location (*Europe*, *Africa*) or a profession (*Smith*, *Porter*). Also, names of companies are often chosen after the name of some of their founders. The problem is that a NER typically only assigns one label to a specific piece of text. In order to increase recall, and given that NE recognition is not an end task, it is therefore theoretically advisable to allow to return multiple labels and then let further modules of the QA system do the final filtering to detect the exact answer. This is the hypothesis that we want to test in the present study. The evaluations presented in this paper include a NER that assigns single labels and a variation of the same NER that produces multiple, overlapping labels.

3 Question Answering

QA systems typically take a question presented by the user posed in natural language. This is then analysed and processed. The final result of the system is an *answer*, again in natural language, to the question of the user. This is different from, what is normally considered, information retrieval in that the user presents a complete question instead of a query consisting of search keywords. Also, instead of a list of relevant documents, a QA system typically tries to find an exact answer to the question.

3.1 AnswerFinder

The experiments discussed in this paper have been conducted within the AnswerFinder project (Mollá and van Zaanen, 2005). In this project, we develop the AnswerFinder question answering system, concentrating on shallow representations of meaning to reduce the impact of paraphrases (different wordings of the same information). Here, we report on a sub-problem we tackled within this project, the actual finding of correct answers in the text.

The AnswerFinder question answering system consists of several phases that essentially work in a sequential manner. Each phase reduces the amount of data the system has to handle from then on. The advantage of this approach is that progressive phases can perform more "expensive" operations on the data.

The first phase is a document retrieval phase that finds documents relevant to the question. This greatly reduces the amount of texts that need to be handled in subsequent steps. Only the best n documents are used from this point on.

Next is the sentence selection phase. From the relevant documents found by the first phase, all sentences are scored against the question. The most relevant sentences according to this score are kept for further processing.

At the moment, we have implemented several sentence selection methods. The most simple one is based on word overlap and looks at the number of words that can be found in both the question and the sentence. This is the method that will be used in the experiments reported in this paper. Other methods implemented, but not used in the experiments, use richer linguistic information. The method based on grammatical relation (Carroll et al., 1998) overlap requires syntactic analysis of the question and the sentence. This is done using the Connexor dependency parser (Tapanainen and Järvinen, 1997). The score is computed by counting the grammatical relations found in both sentence and question. Logical form overlap (Mollá and Gardiner, 2004) relies on logical forms that can be extracted from the grammatical relations. They describe shallow semantics of the question and sentence. Based on the logical form overlap, we have also implemented logical graph overlap (Mollá, 2006). This provides a more fine-grained scoring method to compute the shallow semantic distance between the question and sentence. All of these methods have been used in a full-fledged question answering system (Mollá and van Zaanen, 2006). However, to reduce variables in our experiments, we have decided to use the simplest method only (word overlap) in the experiments reported in this paper.

After the sentence selection phase, the system searches for the exact answers. Some of the sentence selection methods, while computing the distance, already find some possible answers. For example, the logical graphs use rules to find parts of the sentence that may be exact answers to the question. This information is stored together with the sentence. Note that in this article, we are are only interested in the impact of named entity recognition in QA, so we will not use any sentence selection method that finds possible answers.

The sentences remaining after the sentence selection phase are then analysed for named entities. All named entities found in the sentences are considered to be possible answers to the user question.

Once all possible answers to the questions are found, the actual answer selection phase takes place. For this, the question is analysed, which provides information on what kind of answer is expected. This can be, for example, country, river, distance, person, etc. as described in Table 1. The set of possible answers is now considered preferring answers that match the question type.

The best answer (i.e. with the highest score and matching the question type) is returned to the user, which finishes a typical question answering interaction.

4 Named Entity Recognition

The ability of finding exact answers by the AnswerFinder system relies heavily on the quality of the named entity recognition performed on the sentences that are relevant to the user question. Finding all named entities in the sentences is therefore of utmost importance. Missing named entities may mean that the answer to the question cannot be recovered anymore.

We have tried different NERs in the context of question answering. In addition to a general purpose NER, we have developed our own NER. Even though several high quality NERs are available, we thought it important to have full control over the NER to make it better suited for the task at hand.

4.1 ANNIE

ANNIE is part of the Sheffield GATE (General Architecture for Text Engineering) system (Gaizauskas et al., 1996) and stands for "A Nearly-New IE system". This architecture does much more than we need, but it is possible to only extract the NER part of it. Unfortunately, there is not much documentation on the NER in ANNIE. The named entity types found by ANNIE match up with the MUC types as described in Table 2.

ANNIE was chosen as an example of a typical NER because it is freely available to the research community and the named entity types are a subset of the MUC types.

4.2 AFNER

In addition to ANNIE's NER, we also look at the results from the NER that is developed within the AnswerFinder project, called *AFNER*.

4.2.1 General Approach

The NER process used in AFNER consists of two phases. The first phase uses hand-written regular expressions and gazetteers (lists of named entities that are searched for in the sentences). These information sources are combined with machine learning techniques in the second phase.

AFNER first tokenises the given text, applies the regular expressions to each token, and searches for occurrences of the token in the gazetteers. Regular expression matches and list occurrences are used as features in the machine learning classifier. These features are used in combination with token specific features, as well as features derived from the text as a whole. Using a model generated from the annotated corpus, each token is classified as either the beginning of ('B') or in ('I') a particular type of named entity, or out ('OUT') of any named entity. The classified tokens are then appropriately combined into named entities.

4.2.2 First Phase — Regular Expressions and Gazetteers

Regular expressions are useful for finding named entities following identifiable patterns, such as dates, times, monetary expressions, etc. As a result, the entities that can be discovered using regular expressions are limited. However, matching a particular regular expression is a key feature used in identifying entities of these particular types. Gazetteers are useful for finding commonly referenced names of people, places or organisations, but are by no means exhaustive. The purpose of combining lists with other features is to supplement the lists used.

4.2.3 Second Phase — Machine Learning

The second phase involves the machine learning component of AFNER. The technique used is maximum entropy, and the implementation of the classifier is adapted from Franz Josef Och's *YASMET*.[4] The system is trained on the Remedia Corpus (Hirschman et al., 1999), which contains annotations of named entities.

The regular expression and gazetteer matches are used as features, in combination with others

[4] `http://www.fjoch.com/YASMET.html`

pertaining to both individual tokens and tokens in context. Features of individual tokens include those such as capitalisation, alpha/numeric information, etc. Contextual features are those that identify a token amongst surrounding text, or relate to tokens in surrounding text. For example, whether a token is next to a punctuation mark or a capitalised word, or whether a token is always capitalised in a passage of text. Contextual features relating to global information have been used as described by Chieu and Ng (2002). In addition, features of previous tokens are included.

The features are then passed to a maximum entropy classifier which, for every token, returns a list of probabilities of the token to pertain to each category. The categories correspond with each type of entity type prepended with 'B' and 'I', and a general 'OUT' category for tokens not in any entity. The list of entity types used is the same as in the MUC tasks (see Table 2).

Preliminary experiments revealed that often the top two or three entity type probabilities have similar values. For this reason the final named entity labels are computed on the basis of the top n probabilities (provided that they meet a defined threshold), where n is a customisable limit. Currently, a maximum of 3 candidate types are allowed per token.

Classified tokens are then combined according to their classification to produce the final list of named entities. We have experimented with two methods named *single* and *multiple*. For single type combination only one entity can be associated with a string, whereas for multiple type combination several entities can be associated. Also, the multiple type combination allows overlaps of entities. The multiple type combination aims at increasing recall at the expense of ambiguous labelling and decrease of precision.

In the case of multiple type combination (see Figure 1 for an example), each label prepended with 'B' signals the beginning of a named entity of the relevant type, and each 'I' label continues a named entity if it is preceded by a 'B' or 'I' label of the same type. If an 'I' label does not appear after a 'B' classification, it is treated as a 'B' label. In addition, if a 'B' label is preceded by an 'I' label, it will be both added as a separate entity (with the previous entity ending) and appended to the previous entity.

The single type combination (Figure 2) is implemented by filtering out all the overlapping entities of the output of the multiple type combination. This is done by selecting the longest-spanning entity and discarding all substring or overlapping strings. If there are two entities associated with exactly the same string, the one with higher probability is chosen.

The probability of a multi-token entity is computed by combining the individual token probabilities. Currently we use the geometric mean but we are exploring other possibilities. If P_i is the probability of token i and $P_{1...n}$ is the probability of the entire sentence, the geometric mean of the probabilities is computed as:

$$P_{1...n} = e^{\frac{\sum_{i=1}^{n} \log P_i}{n}}$$

5 Results

To evaluate the impact of the quality of NER within the context of question answering, we ran the AnswerFinder system using each of the named entity recognisers, ANNIE, AFNER$_s$ and AFNER$_m$. This section first explains the experimental setup we used, then shows and discusses the results.

5.1 Experimental setup

To evaluate AnswerFinder we used the data available for participants of the QA track of the 2005 TREC competition-based conference[5]. This competition provides us with a nice setting to measure the impact of the NERs. We simply use the documents and questions provided during the TREC 2005 competition. To determine whether a document or text fragment contains the answer we use Ken Litkowsky's answer patterns, also available at the TREC website.

The questions in TREC 2005 are grouped by topic. The competition consisted of 75 topics, with a total of 530 questions. These questions are divided into three different types: factoid, list, and other. In this paper, we only consider the factoid questions, that is, questions that require a single fact as answer. List asks for a list of answers and other is answered by giving any additional information about the topic. There are 362 factoid questions in the question set.

In the experiments, AnswerFinder uses the TREC data as follows. First, we apply docu-

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

BPER	ILOC						
IPER	BLOC			BLOC		BDATE	
BLOC	IPER	OUT	OUT	IPER	OUT	IDATE	OUT
Jack	*London*	*lived*	*in*	*Oakland*	*in*	*1885*	.
PERSON	LOCATION			LOCATION		DATE	
PERSON				PERSON			
LOCATION							

Figure 1: Named entities as multiple labels. The token-based labels appear above the words. The final NE labels appear below the words.

BPER	ILOC						
IPER	BLOC			BLOC		BDATE	
BLOC	IPER	OUT	OUT	IPER	OUT	IDATE	OUT
Jack	*London*	*lived*	*in*	*Oakland*	*in*	*1885*	.
PERSON				LOCATION		DATE	

Figure 2: Named entities as single labels. The token-based labels appear above the words. The resulting NE labels appear below the words.

ment selection (using the list of relevant documents for each question provided by TREC). From these documents, we select the n best sentences based on word overlap between the sentence and the question.

We can now compute an upper-bound baseline. By taking the selected sentences as answers, we can compute the maximum score possible from a question answering perspective. By not requiring exactly matching answers, we can count the number of questions that could be answered if the answer selection phase would be perfect. In other words, we measure the percentage of questions that can still be answered if the answer selection part of the system would be perfect.

Next, we run experiments with the same settings, but applying each of the NERs to the relevant sentences. All named entities that are found in these sentences are then considered possible answers to the question and again the percentage of questions that can be answered is computed.

Finally, we embed the NERs in a simplified version of AnswerFinder to test their impact in a baseline QA system.

# of documents	% of questions
10	75.5%
20	81.6%
30	86.9%
40	89.5%
50	92.1%

Table 3: Percentage of factoid questions that can still be answered after document selection

# of sentences	% of questions
5	42.4%
10	49.9%
20	62.0%
30	65.4%
40	68.8%
50	70.8%
60	73.0%
70	73.7%

Table 4: Percentage of factoid questions that can still be answered after sentence selection from the top 50 documents

5.2 Empirical results

In Table 3 we see the percentage of questions that can still be answered after document selection. The table reflects the intuition that, the smaller the number of preselected documents, the more likely it is that the document that contains the answer is left out. The documents are selected using a list of relevant documents provided for the competition.

If we continue with 50 documents after document selection, we can select relevant sentences from the text in these documents using the word overlap metric. We end up with the percentages as given in Table 4.

There is quite a dramatic drop from 92.1% in all the documents to 73.7% with 70 sentences selected. This can be explained from the fact that the

# of	% of questions		
sentences	ANNIE	AFNER$_s$	AFNER$_m$
5	27.9%	11.6%	27.7%
10	33.0%	13.6%	33.3%
20	41.4%	17.7%	41.9%
30	44.3%	19.0%	45.6%
40	46.2%	19.9%	47.4%
50	47.8%	20.5%	48.8%
60	49.3%	21.3%	51.0%
70	50.5%	21.3%	51.5%

Table 5: Percentage of factoid questions that can still be answered after NE recognition from the top 50 documents

word overlap sentence selection is not extremely sophisticated. It only looks at words that can be found in both the question and sentence. In practice, the measure is very coarse-grained. However, we are not particularly interested in perfect answers here, these figures are upper-bounds in the experiment.

From the selected sentences now we extract all named entities. The results are summarised in Table 5.

The figures of Table 5 approximate recall in that they indicate the questions where the NER has identified a correct answer (among possibly many wrong answers).

The best results are those provided by AFNER$_m$ and they are closely followed by ANNIE. This is an interesting result in that AFNER has been trained with the Remedia Corpus, which is a very small corpus on a domain that is different from the AQUAINT corpus. In contrast, ANNIE is fine-tuned for the domain. Given a larger training corpus of the same domain, AFNER$_m$'s results would presumably be much better than ANNIE's.

The results of AFNER$_s$ are much worse than the other two NERs. This clearly indicates that some of the additional entities found by AFNER$_m$ are indeed correct.

It is expected that precision would be different in each NER and, in principle, the noise introduced by the erroneous labels may impact the results returned by a QA system integrating the NER. We have tested the NERs extrinsically by applying them to a baseline setting of AnswerFinder. In particular, the baseline setting of AnswerFinder applies the sentence preselection methods described above and then simply returns the most frequent entity found in the sentences preselected. If there are several entities sharing the top position then one is chosen randomly. In other words, the baseline ignores the question type and the actual context of the entity. We decided to use this baseline setting because it is more closely related to the precision of the NERs than other more sophisticated settings. The results are shown in Table 6.

# of	% of questions		
sentences	ANNIE	AFNER$_s$	AFNER$_m$
10	6.2%	2.4%	5.0%
20	6.2%	1.9%	7.0%
30	4.9%	1.4%	6.8%
40	3.7%	1.4%	6.0%
50	4.0%	1.2%	5.1%
60	3.5%	0.8%	5.4%
70	3.5%	0.8%	4.9%

Table 6: Percentage of factoid questions that found an answer in a baseline QA system given the top 50 documents

The figures show a drastic drop in the results. This is understandable given that the baseline QA system used is very basic. A higher-performance QA system would of course give better results.

The best results are those using AFNER$_m$. This confirms our hypothesis that a NER that allows multiple labels produces data that are more suitable for a QA system than a "traditional" single-label NER. The results suggest that, as long as recall is high, precision does not need to be too high. Thus there is no need to develop a high-precision NER.

The table also indicates a degradation of the performance of the QA system as the number of preselected sentences increases. This indicates that the baseline system is sensitive to noise. The bottom-scoring sentences are less relevant to the question and therefore are more likely not to contain the answer. If these sentences contain highly frequent NEs, those NEs might displace the correct answer from the top position. A high-performance QA system that is less sensitive to noise would probably produce better results as the number of preselected sentences increases (possibly at the expense of speed). The fact that AFNER$_m$, which produces higher recall than AFNER$_s$ according to Table 5, still obtains the best results in the baseline QA system according to Table 6, suggests that the amount of noise introduced by the additional entities does not affect negatively the process of extracting the answer.

6 Summary and Conclusion

In this paper we have focused on the impact of introducing multiple labels with the aim to increase recall in a NER for the task of question answering. In our experiments we have tested the impact of the ANNIE system, and two variations of AFNER, our custom-built system that can be tuned to produce either single labels or multiple labels. The experiments confirm the hypothesis that allowing multiple labelling in order to increase recall of named entities benefits the task of QA. In other words, if the NER has several candidate labels for a string (or a substring of it), it pays off to output the most plausible alternatives. This way the QA system has a better chance to find the answer. The noise introduced by returning more (possibly wrong) entities is offset by the increase of recall.

Further work includes the evaluation of the impact of multi-label NE recognition on higher-performance QA systems. In particular we plan to test various versions of the complete AnswerFinder system (not just the baseline setting) with each of the NERs. In addition, we plan to retrain AFNER using more data and more relevant data and explore the impact of the single and multiple methods on the resulting higher-performance NER.

Acknowledgements

This work is supported by the Australian Research Council under the ARC Discovery grant DP0450750.

References

[Armour et al.2005] Quintin Armour, Nathalie Japkowicz, and Stan Matwin. 2005. The role of named entities in text classification. In *Proceedings CLiNE 2005*, Gatineau, Canada.

[Carroll et al.1998] John Carroll, Ted Briscoe, and Antonio Sanfilippo. 1998. Parser evaluation: a survey and a new proposal. In *Proc. LREC98.*

[Chieu and Ng2002] Haoi Leong Chieu and Hwee Tou Ng. 2002. Named entity recognition: A maximum entropy approach using global information. In *Proceedings COLING 2002.*

[Gaizauskas et al.1996] Robert Gaizauskas, Hamish Cunningham, Yorick Wilks, Peter Rodgers, and Kevin Humphreys. 1996. GATE: an environment to support research and development in natural language engineering. In *Proceedings of the 8th IEEE International Conference on Tools with Artificial Intelligence*, Toulouse, France.

[Hirschman et al.1999] Lynette Hirschman, Marc Light, Eric Breck, and John D. Burger. 1999. Deep Read: A reading comprehension system. In *Proc. ACL'99*. University of Maryland.

[Humphreys et al.2000] Kevin Humphreys, George Demetriou, and Robert Gaizauskas. 2000. Two applications of information extraction to biological science journal articles: Enzyme interactions and protein structures. In *Proceedings of the Pacific Symposium on Biocomputing' 00 (PSB'00)*, pages 502–513. Honolulu, Hawaii.

[Li and Roth2002] Xin Li and Dan Roth. 2002. Learning question classifiers. *Proc. COLING 02*.

[Mollá and Gardiner2004] Diego Mollá and Mary Gardiner. 2004. Answerfinder - question answering by combining lexical, syntactic and semantic information. In Ash Asudeh, Cécile Paris, and Stephen Wan, editors, *Proc. ALTW 2004*, pages 9–16, Sydney, Australia. Macquarie University.

[Mollá and van Zaanen2005] Diego Mollá and Menno van Zaanen. 2005. Learning of graph rules for question answering. In Tim Baldwin and Menno van Zaanen, editors, *Proc. ALTW 2005*. ALTA.

[Mollá and van Zaanen2006] Diego Mollá and Menno van Zaanen. 2006. Answerfinder at TREC 2005. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proc. TREC 2005*. NIST.

[Mollá2006] Diego Mollá. 2006. Learning of graph-based question answering rules. In *Proc. HLT/NAACL 2006 Workshop on Graph Algorithms for Natural Language Processing*, pages 37–44.

[Noguera et al.2005] Elisa Noguera, Antonio Toral, Fernando Llopis, and Rafael Muñoz. 2005. Reducing question answering input data using named entity recognition. In *Proceedings of the 8th International Conference on Text, Speech & Dialogue*, pages 428–434.

[Sundheim1995] Beth M. Sundheim. 1995. Overview of results of the MUC-6 evaluation. In *Proc. Sixth Message Understanding Conference MUC-6*. Morgan Kaufmann Publishers, Inc.

[Tapanainen and Järvinen1997] Pasi Tapanainen and Timo Järvinen. 1997. A non-projective dependency parser. In *Proc. ANLP-97*. ACL.

[Voorhees1999] Ellen M. Voorhees. 1999. The TREC-8 question answering track report. In Ellen M. Voorhees and Donna K. Harman, editors, *Proc. TREC-8*, number 500-246 in NIST Special Publication. NIST.