A Named Entity Recogniser for Question Answering

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

Abstract

Named Entity Recognisers (NERs) are typically used by question answering (QA) systems as means to preselect answer candidates. However, there has not been much work on the formal assessment of the use of NERs for QA nor on their optimal parameters. In this paper we investigate the main characteristics of a NER for QA. The results show that it is important to maintain high recall to retain all possible answers on the one hand, while high precision is essential during the final answer selection phase. We present an NER designed for QA, which aims at having a high recall.

1 Introduction

Named Entity (NE) recognition is the task of finding instances of specific types of entities in free text and is performed by a Named Entity Recogniser (NER). Typical entity types are person or company names, dates and times, and distances. Firstly defined as a separate task in the Message Understanding Conferences (Sundheim, 1995), NE recognition is currently used in a varied range of applications, such as 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 NE recognition for text-based question answering (question answering or QA henceforth). There has been a major 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 Microsoft[3].

A major part of the current research in QA focuses on finding fact-based answers. Often answers to these factoid questions are NEs. For this reason most QA systems incorporate a NER to detect answer candidates that are processed by subsequent modules. The positive impact of NE recognition in QA is widely acknowledged and there are studies that confirm this (Noguera et al., 2005, for example). However, there is no formal study of the optimal characteristics of a NER within the context of QA. Most often a NER is used off-the-shelf, without any fine tuning and usually it is treated as a black box that has been developed independently of the task of QA.

This paper is one step towards a formal study of the impact of NEs in a QA system. In particular, section 2 comments on the desiderata of a NER for QA. Next, section 3 describes the QA framework on which a selection of NERs will be evaluated. Section 4 introduces the NERs under evaluation, with special emphasis on AFNER, a NER designed specifically for QA. Section 5 presents the results of various experiments evaluating the NERs, and finally section 6 presents the concluding remarks and lines of further research.

2 Named Entity Recognition for Question Answering

Within the context of QA, a NER is used in two ways: to filter out strings (such as sentences) that do not possibly contain the answer, or to find possible exact answers. In the first case, the type of the expected answer is determined during the question analysis stage and this type is mapped to a list of named entity types. The NER is then used to single out the named entity types appearing in a text fragment. If a piece of text does not contain

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

any named entity with a type compatible with the type of the expected answer, the text is discarded or heavily penalised.

In the second case, the NER is applied to a string and the found NEs are considered to be possible answers. Once all NEs are found, the expected answer type that was found during the question analysis phase (together with other information) is used to select the NE that best fits the question. This is the case we will be mainly focussing on in this article.

A NER developed for a generic NE recognition task (or for information extraction) is typically fine-tuned for a good balance between recall and precision. In this paper, however, we investigate whether this is what is needed in the context of QA. We expect that for QA a NER that concentrates on high recall and lower precision performs better than one with high precision but low recall. The rationale is that with high recall, even though more incorrect potential answers are passed on to the next phase after NER, the score of these wrong answers can be lowered using other means in subsequent modules. However, if a correct answer does not pass the NER filter (which happens with low recall) the answer will never be found. The results shown in this article, however, illustrate that it is not as simple as this.

To measure the impact of recall in a QA context, we present a newly developed NER that concentrates on providing recall that is higher than that of other freely available NERs. We will then test the impact of recall of NERs in a simple QA system.

3 Question Answering

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 using 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 computationally 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 text 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.[4]

At the moment, we have implemented several sentence selection methods which are described in Molla et al. (2007). To reduce variables in the experiments reported in this paper, we have decided to use the simplest selection method only, which is based on word overlap. Sentences are scored based on the number of words that can be found in both the question and the sentence. Other methods implemented, but not used in the experiments, use richer linguistic information.

The sentences remaining after the sentence selection phase are then handed to a NER. All NEs found in these sentences are considered to be possible answers to the user question. Since we want to measure the impact of the NER in the QA system, we only use the NEs as possible answers. All other mechanisms of finding answers that are otherwise used in AnswerFinder are not used in these experiments.

In these experiments we do not use NEs to filter out sentences (the first case in section 2). Currently we are investigating possible ways to use the NE information for sentence selection.

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. Using this information, the possible answers that match the expected answer type are selected and scored.

Finally, the best answer is returned to the user. Best answer in this context is considered as the answer with both the highest score and matching the answer type or simply the answer with the high-

[4] Selecting sentences out of context assumes that the answer to a question can be found in one sentence. At the moment, the system does not handle answers that are distributed over multiple sentences. Similarly, the system expects that the answer can be found in one document. This does not allow multi-document summaries as requested in the DUC summarisation competitions (Dang, 2006).

Class	Type	# in BBN
ENAMEX	Organization	30,248
	Person	13,751
	Location	14,656
TIMEX	Date	20,672
	Time	1,069
NUMEX	Money	11,097
	Percent	5,976

Table 1: Entities used in the MUC tasks and number of occurrences in the BBN corpus

est score if none of the possible answers fits the answer type.

4 Named Entity Recognition

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 NE types found by ANNIE match up with the MUC types as described in Table 1.

ANNIE was chosen as an example of a typical NER because it is freely available to the research community and the NE types match with the MUC types.

4.2 AFNER

In addition to the ANNIE NER, we also looked at the results from the NER that is developed within the AnswerFinder project, called *AFNER*, which is based on machine learning. The technique used is maximum entropy, and the implementation of the classifier is adapted from Franz Josef Och's *YASMET*.[5] The system is trained on the BBN Pronoun Coreference and Entity Type Corpus (the BBN corpus in short), which is available at the Linguistics Data Consortium[6]. The NE hierarchy of the BBN corpus uses a much finer gradation than that of ANNIE. To enable a comparison between AFNERand ANNIE we have mapped the BBN corpus entities to the MUC set. Table 1 contains the number of entities in the BBN corpus after they have been mapped to the MUC set.

4.2.1 Features

The features used by AFNER combine regular expressions and gazetteers with properties internal and external to the token.

Regular expressions are useful for identifying specific patterns characteristic of some entity types such as dates, times, speed and monetary expressions. The range of entities that can be discovered using regular expressions is limited, but the precision of the regular expressions is high. Therefore, 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. If an expression is found in one of the lists, then it is likely that the expression is of the type indicated by the list, but sometimes this is not the case. By introducing gazetteers as additional features in the classifier it becomes possible to use other features that may be more determinant for the categorisation of a specific token in particular cases. AFNER uses three lists (locations, person names, and organisations), with a total of about 55,000 list items.

Features relating internal token properties include those such as capitalisation, alpha/numeric information, etc. and are listed in Table 2.

In addition AFNER incorporates contextual features. These are features that identify a token amongst surrounding text, or relate a token to tokens in surrounding text. In particular AFNER applies a set of regular expressions to the neighbouring tokens within a context window and records the match results as features. These regular expressions detect patterns such as whether the neighbouring token is made of two digits, or whether it is a currency name. Additional features include the class assigned to the previous token and all of its class probabilities.

A final set of features relates to global information inspired on those features described by Chieu and Ng (2002). Currently AFNER only checks

[5] http://www.fjoch.com/YASMET.html

[6] http://www.ldc.upenn.edu

Regular Expressions	Specific patterns for dates, times, etc
FoundInList	The token is a member of a gazetteer
InitCaps	The first letter is a capital letter
AllCaps	The entire word is capitalised
MixedCaps	The word contains upper case and lower case letters
IsSentEnd	The token is an end of sentence character
InitCapPeriod	Starts with capital letter and ends with period
OneCap	The word is a single capitalised letter
ContainDigit	The word contains a digit
NumberString	The word is a number word ('one', 'thousand', etc.)
PrepPreceded	The word is preceded by a preposition (in a window of 4 tokens)
PrevClass	The class assigned to the previous token
ProbClass	The probability assigned to a particular class in the previous token
AlwaysCapped	The token is capitalised every time it appears

Table 2: Features used in AFNER

whether a token is always capitalised in a passage of text.

4.2.2 General Method

The features are passed to a maximum entropy classifier which, for each 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' (indicating whether the token is the begin or inside a NE respectively), 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 1).

Preliminary experiments revealed that often the top two or three entity type probabilities have similar values. For this reason the final NE labels are computed on the basis of the probabilities that are higher than a threshold. The threshold is relative to the highest probability associated to the token.

By allowing tokens to have multiple tags assigned to it, AFNER aims at high recall. The presence of multiple tags means that NEs can be nested (AFNER may output NEs that are contained in other NEs). In the rest of the article, we allow AFNER to classify two tags at most for each token. Preliminary experiments have shown that this greatly increases recall, whereas using three tags for each token only increases recall a bit more while decreasing precision massively.

Classified tokens are then combined according to their classification to produce the final list of NEs. The general method is as follows. Each label prepended with 'B' signals the beginning of a possible NE of the relevant type and each 'I' label continues a NE 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 result is a set of tags that may overlap (Figure 1). This is filtered 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 (Figure 2).

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 $P_{1\ldots n}$ is the probability of the entire sentence, the geometric mean of the probabilities is computed as:

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

5 Results

To measure the impact of the quality of NERs and the quantity of the NEs generated in the context of question answering, we incorporated the two NERs in a question answering system. We first explain the experimental setup, followed by empirical results and a discussion.

5.1 Experimental setup

In this evaluation, we use the data provided for the question answering track of the 2005

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.

TREC competition-based conference[7]. The data is provided specifically to measure performance of question answering systems and as such it provides a good benchmark set.

The document collection consists of just over one million news paper articles. Additionally, a list of 530 questions, grouped by topic, is provided. Each question can be one of three types: factoid (requiring a single fact as an answer), list (which may have multiple answers) and other (asking for any additional information on a topic). Here, we only consider the factoid questions, of which there are 362. To determine whether a question is answered correctly, we use Ken Litkowsky's answer patterns, which are also available from the TREC website.[8] For each topic, a list of relevant documents was generated by running the PRISE information retrieval engine on the question targets. This list of documents is also available from the TREC website. For each target, 1,000 documents were provided in the same order as returned by PRISE.

The question answering system used in the experiments is a simplification of AnswerFinder and works as follows. Firstly, the system selects documents that are relevant to a particular question using the list of relevant documents. From these documents, the best sentences are selected based on the size of the word overlap between the sentence and the question, where larger overlaps are better. The selected sentences are then provided as input for the different NERs, ANNIE and AFNER. This results in a list of NEs that the system takes as possible answers. Finally, the system selects only those NEs that match with the type of question and the NE with the highest score (which is the number of times the NE was found) is returned. If multiple NEs have the same score, one is selected at random.

The impact of each of the phases is computed by considering the percentage of questions that can still be answered after that phase. Additionally, we will provide information on the amount of text that is left with respect to the previous phase (as each phase removes unnecessary text).

5.2 Empirical results

Table 3 contains the results after document selection. The documents are selected according to the output of PRISE as described above. In particular we used the list of preselected documents provided with the TREC 2005 questions. The results presented here are computed by taking the top *n* documents from the list (where *n* is 10, 20, etc.).

The percentages indicate how many of all questions can still be answered, i.e. in how many of the questions the answer can still be found in the text that is left. Obviously, increasing the number of retained documents increases the percentage. This percentage is an upper-bound for the rest of the

[7] `http://trec.nist.gov`

[8] We assume a question is answered correctly if the answer can be identified as a string in the document collection. We do not check whether the information in the context of the answer actually supports the string to be the correct answer.

# of documents	% of questions
10	75.5%
20	81.6%
30	86.9%
40	89.5%
50	92.1%
100	93.6%
500	95.6%
1,000	95.9%

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

# of sentences	% of questions	% of remaining text
10	49.9%	(2.4%)
20	62.0%	(3.7%)
30	65.4%	(4.9%)
40	68.8%	(6.1%)
50	70.8%	(7.1%)
60	73.0%	(8.1%)
70	73.7%	(8.9%)
80	75.0%	(9.8%)
90	76.0%	(10.6%)
100	76.2%	(11.4%)

Table 4: Percentage of factoid questions that can still be answered after selecting sentences from the top 50 documents and percentage of remaining text in brackets

phases. Note that even when all 1,000 documents selected by PRISE are used, not all questions can be answered.

We will continue with the top 50 documents after document selection. Considering all sentences in these documents, we only select the best *n* according to the word overlap metric. This metric counts the number of words that can be found in both question and sentence, not counting function words. The results after sentence selection can be found in Table 4. The percentages in brackets show the percentage of text that is retained with respect to the full amount of text of the top 50 documents.

It is interesting to see that there is a drop from 92.1% answerable questions to 76.2% when selecting the best 100 sentences. However, this loss of answers can be expected, since we are only left with only 11.4% of the initial text. The relatively large drop in results can also be explained from the fact that the word overlap metric is crude.

# of	% of questions			
sent.	ANNIE		AFNER	
10	34.4%	(14.4%)	37.0%	(22.5%)
20	42.8%	(14.9%)	45.9%	(23.0%)
30	45.9%	(15.0%)	50.0%	(23.1%)
40	47.8%	(14.9%)	52.3%	(23.0%)
50	49.3%	(15.0%)	53.7%	(23.1%)
60	50.8%	(15.0%)	55.1%	(23.1%)
70	52.0%	(15.0%)	55.9%	(23.1%)
80	52.6%	(15.1%)	57.0%	(23.2%)
90	52.8%	(15.2%)	58.2%	(23.3%)
100	52.8%	(15.2%)	58.2%	(23.4%)

Table 5: Percentage of factoid questions that can still be answered after NE recognition of the selected sentences from the top 50 documents and the percentage of text that is retained (in brackets)

The scores given to the sentences are very coarse-grained and this phase can probably be improved using more sophisticated metrics. However, these figures serve as upper-bounds in the experiment.

The selected sentences are now handed to the two NERs. These find a list of NEs, which are taken as possible answers. Table 5 illustrates the percentage of questions that can still be answered together with the percentage of text remaining with respect to the amount of text in the selected sentences.

It has to be taken into account that there is an upper bound of questions that can be answered correctly that is given in Table 4. However, not all questions can be answered using NEs, so the actual upper bound is a little bit lower.

The percentages of answerable questions given in Table 5 approximate recall. They indicate how many correct answers are identified by the NERs, whereas the percentages in brackets (again these are percentages of text retained with respect to the selected sentences) give a rough indication of (inverted) precision in that small percentages would ideally contain the correct entities only, whereas larger percentages contain more noise. The aim is therefore to reduce the text as much as possible while retaining as many answerable questions as possible.

Table 5 shows that AFNER generates most potential answers (23.4% of the text is retained when using 100 sentences compared to ANNIE with 15.2%) and it also covers most questions. This means that AFNER has a higher recall, but a lower precision. ANNIE has a lower recall (it answers 52.8% of the questions with 100 sentences), but retains only 15.2% of the text.

The results provided in Table 5 illustrate that it is important to have a high recall. In a full-blown QA system, NEs found by ANNIE can used to answers less questions that those found by AFNER. On the other hand, it may be the case that the additional text AFNER introduces works as noise that degrades the final results of the question answering system.

To measure the impact of additional text versus percentage of answerable questions, we test the different NERs in our simplified version of AnswerFinder. The NEs found as described above are filtered by removing those NEs that are not of the expected answer type. From the remaining NEs, the one that occurs most often in the text is selected. If there are several with the same score (say *n*), we count the number of those answers that are correct and divide them by *n*. The results of this system are shown in Table 6.

# of	% of questions	
sentences	ANNIE	AFNER
10	8.3%	6.0%
20	8.4%	6.7%
30	9.3%	6.5%
40	8.6%	7.3%
50	8.1%	7.0%
60	7.6%	7.3%
70	7.5%	6.8%
80	7.9%	6.4%
90	9.0%	6.5%
100	8.5%	6.1%

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

With respect to the results in Table 5, the figures in Table 6 are drastically lower. This illustrates that the final answer selection method is very crude. More sophisticated answer selection should improve on these results.

There does not seem to be a consistent line in each of the system's results. This indicates that both systems are very sensitive to the actual counts of the NEs they find in the sentences. Using more data clearly does not mean that better answers are found.

The results in Table 6 show that the advantage AFNER has in finding more answers to questions is reduced when the output is used directly in a QA system. The amount of noise introduced by increasing the recall has a negative effect on the system output.

6 Summary and Conclusion

In this paper we have introduced a NER developed within the context of QA. In our experiments we have tested the impact of the NERs available in ANNIE and AFNER, our custom-built system. The experiments show that AFNER's recall is higher than ANNIE's. The results also showed that there is a complex interaction between the effects of recall and performance when using NEs in a QA context. Whereas recall is important to be able to answer as many questions as possible, the amount of noise introduced to do this should be kept at a minimum.

A more detailed analysis of the impact of NEs quality in a QA system is needed. The means of selecting answers in the system described here is very crude and is highly dependable on the quality of the NEs to start with. Additional information that can be extracted during the QA system run should be able to identify correct answers better. In particular, we plan to test various versions of the complete AnswerFinder system (not just the baseline setting) with both NERs.

Even though the recall of AFNER is higher than that of ANNIE, the precision is probably too low. In Table 5 AFNER generated almost twice the amout of output of ANNIE. Part of this is noise that resulted in reduced scores in the final system. We are currently studying methods to increase the precision of AFNER while retaining the high recall, in order to build a NER that is better suited to QA.

Acknowledgements

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

Special acknowledgements are due to Daniel Smith and Alex Chilvers, who were involved in the development of AFNER.

References

Quintin Armour, Nathalie Japkowicz, and Stan Matwin. 2005. The role of named entities in text classification. In *Proceedings CLiNE*

2005. Gatineau, Canada. URL http://www.crtl.ca/cline05/cline05_papers/ArmourJapkowiczMatwin.pdf.

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

Hoa Tran Dang. 2006. Duc 2005: Evaluation of question-focused summarization systems. In *Proceedings of the Workshop on Task-Focused Summarization and Question Answering*, pages 48–55. Association for Computational Linguistics, Sydney. URL http://www.aclweb.org/anthology/W/W06/W06-0707.

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. URL http://www.dcs.shef.ac.uk/research/groups/nlp/gate/.

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. URL http://www.bionlp.org/psb2000/humphreys.pdf.

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. URL http://www.alta.asn.au/.

Diego Molla, Menno van Zaanen, and Luiz Pizzato. 2007. Answerfinder at trec 2006. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proceedings TREC 2006*, page 8 pages.

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. URL http://dx.doi.org/10.1007/11551874_55.

Beth M. Sundheim. 1995. Overview of results of the MUC-6 evaluation. In *Proc. Sixth Message Understanding Conference MUC-6*. Morgan Kaufmann Publishers, Inc. URL http://acl.ldc.upenn.edu/M/M95/M95-1002.pdf.

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. URL http://trec.nist.gov/pubs.html.