Improving Passage Retrieval in Question Answering Using NLP

Jörg Tiedemann

Alfa Informatica, University of Groningen,
Groningen, The Netherlands
j.tiedemann@rug.nl
http://www.let.rug.nl/~tiedeman

Abstract. This paper describes an approach for the integration of linguistic information in passage retrieval in an open-source question answering system for Dutch. Annotation produced by the wide-coverage dependency parser Alpino is stored in multiple index layers to be matched with natural language question that have been analyzed by the same parser. We present a genetic algorithm to select features to be included in retrieval queries and for optimizing keyword weights. The system is trained on questions annotated with their answers from the competition on Dutch question answering within the Cross-Language Evaluation Forum (CLEF). The optimization yielded a significant improvement of about 19% in mean reciprocal rank scores on unseen evaluation data compared to the base-line using traditional information retrieval with plain text keywords.

1 Introduction

Question Answering (QA) systems aim at locating answers to natural language questions in large document collections. This is usually achieved using a combination of Information Extraction (IE) and Information Retrieval (IR) components. Natural Language Processing (NLP) and linguistic resources are frequently used in QA systems, see e.g. [1,2], although not very often for the retrieval component (some exceptions are [3,4,5]). Using NLP in information retrieval has been the goal for many researchers. However, it has been argued that NLP tools are still to brittle and inefficient to be used in information retrieval [4]. Several experiments using NLP techniques have been reported in the literature, e.g. lemmatization and compound splitting [6,7,8,9,10], query term selection and weighting [11,12], extraction of noun phrases and other linguistically motivated units [13,14,3]. However, most of the studies using deep linguistic analyses resulted in only little success or even decreasing performance, see, e.g., [15,16]. Simple techniques such as stemming and stop word removal seem to be much more effective than more sophisticated techniques at least for languages with relatively poor morphological variation. However, in [4], the authors show that syntactic analyses can be very useful for retrieval performance in QA when selected carefully. They argue that NLP technology should only be used in cases where we know that

C. Bento, A. Cardoso, and G. Dias (Eds.): EPIA 2005, LNAI 3808, pp. 634–646, 2005.

they are helpful without abandoning simpler techniques. Along these lines we like to use deep syntactic analyses in the retrieval component of our QA system in such a way that we select features and feature combinations that have shown to improve the performance. In contrast to [4], we do not investigate selected linguistic phenomena but a whole spectrum of natural language questions as defined by the Cross-Language Evaluation Forum (CLEF) in their question answering track. This paper describes an iterative learning approach for feature selection and query optimization.

The next section includes a brief description of our retrieval component in our question answering system. Thereafter the query optimization algorithm is described followed by experimental results using questions from the CLEF competitions on Dutch QA.

2 Question Answering with Dependency Relations

In our investigations we focus on open-domain question answering for Dutch. The system we are building, [17], consists of two streams: a table look-up strategy using off-line information extraction and an on-line strategy using passage retrieval and on-the-fly answer extraction. In both strategies we use syntactic information produced by a wide-coverage dependency parser for Dutch, Alpino [18]. In the off-line strategy we use syntactic patterns to extract information from unrestricted text to be stored in fact tables [19]. For the on-line strategy, we assume that there is a certain overlap between syntactic relations in the question and in passages containing the answers. Hence, the entire document collection has to be parsed to apply syntactic patterns for off-line information extraction and to match questions to possible answers. The corpus provided by CLEF contains about 1.1 million paragraphs that include altogether about 4 million sentences. They have been parsed by Alpino and stored in XML tree structures (about 0.35% of the sentences could not be analyzed because of parsing timeouts). Incoming questions are parsed in the same way. The system uses the analyzed questions to determine the question type and to formulate a query to the information retrieval component.[1] The parse tree and the question type are then used to locate possible answers in passages retrieved by IR. Information retrieval is used to reduce the search space for the answer extraction components to make it feasible to run on-line QA. Hence, the system relies on the passages retrieved by this component and fails if IR does not provide relevant documents containing answers.

Traditional IR uses a bag-of-word approach using plain text keywords to be matched with word-vectors describing documents. The result is a ranked list of documents. This approach will be our base-line (including Dutch stemming and stop word removal). In our system, we integrated an interface to several off-the-shelf IR engines. Here, we will use Lucene from the Apache Jakarta project [20].

[1] The table look-up strategy is used if the question type matches a table in the fact databases extracted off-line. Information retrieval and the on-line strategy is used if table look-up fails.