BioGrapher: Biography Questions as a Restricted Domain Question Answering Task

Oren Tsur
Text and Data Mining Group
Bar Ilan University
tsuror@cs.biu.ac.il

Maarten de Rijke
Informatics Institute
University of Amsterdam
mdr@science.uva.nl

Khalil Sima'an
Institute for Logic, Language and Computation
University of Amsterdam
simaan@science.uva.nl

Abstract

We address Question Answering (QA) for biographical questions, i.e., questions asking for biographical facts about persons. The domain of biographical documents differs from other restricted domains in that the available collections of biographies are inherently incomplete: a major challenge is to answer questions about persons for whom biographical information is not present in biography collections. We present BioGrapher, a biographical QA system that addresses this problem by machine learning algorithms for biography classification. BioGrapher first attempts to answer a question by searching in a given collection of biographies, using techniques tailored for the restricted nature of the domain. If a biography is not found, BioGrapher attempts to find an answer on the web: it retrieves documents using a web search engine, filters these using the biography classifier, and then extracts answers from documents classified as biographies. Our empirical results show that biographical classification, prior to answer extraction, improves the results.

1 Introduction

Although most current research in question answering (QA) is oriented towards open domains, as witnessed by evaluation exercises such as TREC, CLEF, and NTCIR, various significant applications concern restricted domains, e.g., software manuals. In restricted domains, a QA system faces questions and documents that exhibit less variation in language use (e.g., words and fixed phrases, more specific terminology) than in an open domain, and it could access high-quality knowledge sources that cover the entire domain. Open domain QA as it is assessed at TREC, CLEF, and NTCIR concerns a broad variety of fairly restricted question types, such as location questions, monetary questions, biography questions, questions that ask for concept definitions, etc. How useful or effective is it to adopt a restricted domain approach to some of these question types? In this paper we explore so-called biographical questions, e.g., *Who was Algar Hiss?* or *Who is Sir John Hale?*, i.e., questions that demand answers consisting of biograpical key facts that are typically found in biographies and that typically involve fixed phrases about, e.g., birthdates, education, societal roles. This type of questions was found to be quite frequent in search engine logs. We believe that biographical questions can be usefully viewed as defining a restricted domain for QA: the domain of biographical information as represented by biographies.

Ideally, biographical questions are answered by retrieving a biography from some existing collection of biographies (such as `biography.com`) and extracting snippets from it. Such resources, however, have a limited coverage. There will always be people whose biographical information is not contained in any of the existing collections. This necessitates retrieval of "biography-like" documents, i.e., documents with biographical information. The problem of identifying biography-like documents by machine learning algorithms turns out to be a challenging but rewarding task as we will see below.

In this paper we address the problem of question answering within the biographical domain. We describe BioGrapher, a restricted domain QA system for answering bibliographical questions in which a baseline approach, that exploits biography collections, is extended with a trainable biography classifier operating on the web in order to enhance coverage. The baseline system helps us understand the usefulness of existing high quality biography collections within a QA system. The extension of our baseline approach concerns the problem of identifying biography-like documents, and the extraction from such documents of answers for questions that could not be answered using biography collections. A main challenge lies in constructing an algorithm for identifying documents containing usefull biographical information that may provide an answer for a given question. To addresss this challenge, we explore two machine learning algorithms: Ripper

(Cohen and Singer, 1996) and Support-Vector Machines (Joachims, 1998).

Section 2 provides some background, and in Section 3 we briefly describe our baseline QA system, based on external knowledge sources complemented with a naive approach to retrieving biography snippets using a web search engine. In Section 4 we prepare the ground for our text classification experiments. In Section 5 we present two classifiers: one loosely based on the Ripper algorithm and the other based on an SVM classifier. In Section 6 we compare the performanc of the baseline against versions of the system integrated with the two classifiers. Section 7 discusses the results and considers the possibility of applying our approach to other restricted domains. We conclude in Section 8.

2 Related Work

Two kinds of related work are relevant to this paper: question answering against external knowledge sources, and genre detection (using classifiers). We briefly discuss both.

Many QA research groups employed External Knowledge Sources in order to improve performance. For instance, (Chu-Carroll and Prager, 2002) used WordNet to answer *what is* questions, using the *isa* hierarchy supported by WordNet. (Hovy et al., 2002; Lin, 2002) used dictionaries such as WordNet and web search results to re-rank answers. (Yang et al., 2003) preformed structure analysis of the knowledge obtained from WordNet and the Web in order to further improve performance.

We refer to (Sebastiani, 2002) for extensive review about machine learning in automated text classification. (Lewis, 1992) were among the first to use machine learning for genre detection trying to categorize *Reuters* articles to predefined categories. Probabilistic classifiers were used by many groups (Lewis, 1998). Much current text classification research is focused on Support Vector Machines, first used for genre detection by (Joachims, 1998).

3 A Naïve Baseline

In this section we describe our baseline QA system. This system was used at TREC 2003, to produce answers to so-called person definition questions (Jijkoun et al., 2004; Voorhees, 2004). We present the results and give a short analysis of the system's performance; as we will see, this provides further motivation for the use of text classification for identifying biography-like documents.

Definition questions at TREC 2003

The QA track at TREC 2003 featured a subtask devoted to definition questions. The latter came in three flavors: person definitions (e.g., *Who is Colin Powell?*), organization definitions (e.g., *What is the U.N.?*), and concept definitions (e.g., *What is goth?*). Here, we are only interested person definitions.

In response to a definition question, systems had to return an unordered set of snippets; each snippet was supposed to be a facet in the definition of the target. There were no limits placed on either the length of an individual answer string or on the number of snippets in the list, although systems were penalized for retrieving extraneous information.

As our primary strategy for handling person definition questions, we consulted external resources. The main resource used is `biography.com`. While such resources contain biographies of many historical and well-known people, they often lack biographies of contemporary people that are not too well-known. To be able to deal with such cases we backed-off to using a web search engine (Google), and applied a naïve heuristic approach. We handcrafted a set of features (such as "born", "graduated", "suffered", etc.) that we felt would trigger for biography-like snippets. Various subsets of the large feature set, together with the target of the definition question, were combined to form queries for the web search engine.

Given a set of candidate answer snippets, we performed two filtering steps before presenting the final answer: we separated non-relevant snippets from valuable snippets and we identified semantically-close snippets. We addressed the first step by analyzing the distances between query terms submitted to the search engine and the sets of features, and by means of shallow syntactic aspects of the different features such as sentence boundaries. To address the second step we developed a snippet similarity metric based on stemming, stopword removal and keyword overlap by sorting and calculating the *Levenshtein distance* measure of similarity.[1]. An example of the snippets filtering can be found in Table 1. The table presents 3 of the returned snippets for the question *Who is Sir John Hale?*. The first and third snippet are filtered out, the first one for non-relevancy and the third for its semantic similarity with the second, shorter, snippet.

[1]The Levenshtein measure is a measure of the similarity between two strings, which are refered to as the source string *s* and the target string *t*. The distance is the number of deletions, insertions, or substitutions required to transform *s* into *t*

1	**Sir** Malcolm Bradbury (**writer**/teacher) Dead. Heart trouble. ... Heywood **Hale** Broun (commentator, **writer**) – **Dead**. **John** Brunner (**author**) **Dead**. Stroke. ... Description: Debunks the rumor of his death...
2	... Professor **Sir John Hale** woke up, had ... For her birthday in 1996, he **wrote** on the ... **John Hale died** in his sleep - possibly following another stroke ...
3	Observer On 29 July 1992, Professor **Sir John Hale** woke up ... her birthday in 1996, he **wrote** on the ... **John Hale died** in his sleep - possibly following another stroke.

Table 1: Snippets filtering for *Who is Sir John Hale?*

Evaluation

Evaluation of individual person definition questions was done using the F-measure: $F = (\beta^2 + 1)P \cdot R)/(\beta^2 P + R)$, where P is precision (to be defined shortly), R is recall (to be defined shortly), and β was set to 5, indicating that precision was more important than recall. Length was used as a crude approximation to precision; it gives a system an allowance of 100 (non-white-space) characters for each correct snippet it retrieved. The precision score is set to one if the response is no longer than this allowance, otherwise it is downgraded using the function $P = 1 - ((length - allowance)/length)$.

As to recall, for each question, the TREC assessors marked some snippets as vital and the remainder as non-vital. The non-vital snippets act as "don't care" condition. That is, systems should be penalized for not retrieving vital nuggets, and for retrieving snippets that are not in the assessors' snippet lists at all, but should be neither penalized nor rewarded for returning a non-vital snippet. To implement the "don't care" condition, snippet recall is computed only over vital snippets (Voorhees, 2004).

In total, 30 person definition questions were evaluated at the TREC 2003 QA track. The overall F score of a run was obtained by averaging over all the individual questions.

Results and Analysis

The F score obtained by the naive system described in this section, on the TREC 2003 person definition questions, was 0.392. An analysis of the results shows that, for questions that could be answered from external biography resources, the baseline system obtains an F score of 0.586.

In post-submission experiments we changed the subsets of features we use in the queries sent to Google as well as the number of queries/subsets we use. The snippet similarity threshold was also tuned in order to filter out more snippets. This resulted in fewer unanswered questions, while the average answer length was decreased as well, by close to 50%. All in all, an informal evaluation showed increase in recall, precision and in the overall F score.

From our experience with our baseline system we learned the following important lesson: having a (relatively) small number of high quality biography sources as a basis for each question's answer is far better than using a broad and large variety of snippets returned by a web search engine. While extending available biography resources so as to seriously boost their coverage is not a feasible option, we want to do the next best thing: make sure we identify good biography-like documents online, so that we can use these to mine snippets from; to this end we will use *text classification*.

4 Preparing for Text Classification

In the previous section we suggested that using a *text classifier* might improve the performance of biography QA. Using text classifiers, we aim to identify biography-like documents from which we can extract answers. In this section we detail the document representations on which we will operate.

Document and Text Representation

Text classifiers represent a document as a set of features $d = \{f_1, f_2, \ldots, f_n\}$ where *n* is the number of *active* features, that is, features that occur in the document. A feature f can be a word, a set of words, a stem or any phrasal structure, depending on its text representation. Each feature has a weight, usually representing the number of occurrences of this feature in the document.

What is a suitable abstract representation of documents for our biography domain? We have defined 7 clusters, groups of words (terms/tokens) with a high degree of pair-wise semantic relatedness. Each cluster has a meta-tag symbol (as can be seen in Table 2) and all occurrences of members of a cluster were substituted by the cluster's meta-tag. An example of a document abstraction can be found in Table 3. This abstraction captures typical similarities between biographical strings; e.g., for the two sentences *John Kennedy was born in 1917* and *William Shakespeare was born in 1564* we get the same abstraction <NAME> <NAME> *was born in* <YEAR>.

It is worth noting that some of the clusters, such as <CAP> and <PLACE>, <CAP> and <PN> and others may overlap. Looking at the example in Table 3, we see that Abbey was born in Chicago, Illinois, but the automatic abstractor misinterpreted the token "Ill.," marking it is <CAP> for capitalized (possibly meaningful) word, but not as

<NAME>	the name of the subject of the biography
<YEAR>	four digits surrounded by white space, probably a year
<D>	sequence of number of digits other than four digits, can be part of a date, age etc.
<CAP>	a capitalized word in the middle of a sentence that wasn't substituted by any other tag
<PN>	a proper name that is not the subject of the biography It substitutes any name out of a list of thousand names
<PLACE>	denotes a name of a place, city or country out of a list of more than thousand places
<MONTH>	denotes one of the twelve months
<NOM>	denotes a nominative

Table 2: Seven meta-tags used for document abstraction

<PLACE>. A similar thing happens with the name "Wooldridge" that is not very common; it should have been <PN> instead of <CAP>.

All procedures described below are preformed on abstract-meta-tagged documents.

5 Identifying Biography Documents

Given a document, the task of a biography classifier is to decide whether a given document is a biography or not. In this section we address the problem of acquiring biography classifiers by training machine learning algorithms on data. We present two biography classification algorithms: a naive classifier based on Ripper (Cohen and Singer, 1996), and another based on SVM (Joachims, 1998). The two methods differ radically both in the way they represent the training data (i.e., document representation), and in their learning approaches. The naive classifier is obtained by a repetitive rule learning algorithm. We modified this algorithm to specifically fit the task of identifying biographies. The SVM learns "linear decision boundaries" between the different classes. We employ here the implementation of SVMs by (Joachims, 1998). Next we discuss the details of how each algorithm was used for learning a biography classifier.

Original	Lincoln, Abbey (b. Anna Marie Wooldridge) 1930 – Jazz singer, composer/arranger, movie actress; born in Chicago, Ill. While a teenager she sang at school and church functions and then toured locally with a dance band.
Abstraction	<NAME>, <NAME> (b . <PN> <CAP> <CAP>) <YEAR> - <CAP> singer , composer/arranger , movie actress ; born in <PLACE> <CAP> . While a teenager <NOM> sang at school and church functions and then toured locally with a dance band .

Table 3: Abstraction of jazz singer Abbey Woolridge's biography

Naive Classifier

We employ this algorithm for its simplicity and scalability. This algorithm learns user-friendly rules, i.e., human-readable conjunctions of propositions, which can be converted to queries for a Boolean search engine. Furthermore, it is known to exhibit relatively good results across a wide variety of classfication problems, including tasks that involve large collections of noisy data, similar to the large document collections that we face in definitional QA. The naive classifier consists of two main stages:

(1) Rules building. This is similar to Ripper's first stage of building an initial rule set. Our algorithm deviates from standard implementations of Ripper in that the terms that serve as the literals in the rules are *n-grams* of various lengths. We feel that n-grams, as opposed to individual literals (as in (Cohen and Singer, 1996), better capture contextual effects, which could be crucial in text classification. Our learner learns the rules as follows. The set of *k*-most frequent n-grams representing the training documents is split into two frequency-ordered lists: *TLP* (term-list-positive) containing the positive example set and *TLN* (term-list-negative) containing the negative examples set. The vector $\vec{w}$ is initialized to be $TLP/(TLP \cap TLN)$, i.e., the most frequent n-grams extracted from the positive set that are not top frequent in the negative set.

(2) Rule optimization. Instead of Ripper's rule pruning stage, our algorithm assigns a weight to each rule/n-gram r in the rules vector according to the formula $\frac{g(n)\cdot f(r)}{C}$, where $g(n)$ is an increasing function in the length of the n-gram (longer n-grams receive higher weights), $f(r)$ is the ratio of the frequency of r in the positive examples to its frequency in the negative examples, and C is the size of the training set. The normalization by C is merely for the purpose of tracking variations of the weights in different sizes of training sets. The preference for longer n-grams can be justified by the intuition that longer n-grams are more informative as they stand for stricter contexts. For example, the string "(<NAME> , <NAME> *born in* <YEAR>" seems more informative than the shorter string *in* <YEAR>).

Training material. The corpus we used as our training set is a collection of 350 biographies. Most of the biographies were randomly sampled from `biography.com`, while the rest were collected from the web. About 130 documents from the New York Times (NYT) 2000 collection were randomly selected as negative example set. The volumes of the positive and negative sets are equal.

Various considerations played a role in building this corpus. The biographies from `biography.com` are 'clean' in the sense that all of them were written as biographies. To enable the learning of features of informal biographies, some other 'noisy' biographies such as biography-like newspaper reviews were added. Furthermore, a small number of different biographies of the same person were manually added in order to enforce variation in style. We also added a small number of biographies from other different sources to avoid any bias towards the biography.com domain.

Validation and tuning. We tuned the naive algorithm on a separate validation set of documents. The validation set was collected in the same way as the training set. It contained 60 biographies, of which 40 were randomly sampled from `biography.com`, 10 'clean' biographies were collected from various online sources, 10 other documents were noisy biographies such as newspaper reviews. In addition, another 40 non-biographical documents were randomly retrieved from the web.

The vector $\vec{w}$ is now used to rank the documents of the validation set V in order to set a threshold θ that minimizes the false-positive and the false-negative errors. Each document $d_j \in V$ in the validation set is represented by a vector $\vec{x}$, where x_i counts the occurrences of w_i in d_j. The score of the document is the normalized inner product of $\vec{x}$ and $\vec{w}$ given by the function $score(d_j) = \frac{\vec{x} \cdot \vec{w}}{length(d_j)}$.

In the validation stage some heuristic modifications were applied by the algorithm. For example, when the person name tag is absent, the document gets the score of zero even though other parameters of the vector may be present. We also normalized document scores by document length.

Support Vector Machines (SVMs)

Now we describe the learning of a biography classifier using SVMs. Unlike many other classifiers, SVMs are capable of learning classification even of non-linearly-separable classes. It is assumed that classes that are non-linearly separable in one dimension may be linearly separable in higher dimension. SVMs offer two important advantages for text classification (Joachims, 1998; Sebastiani, 2002): (1) Term selection is often not needed, as SVMs tend to be fairly robust to overfitting and can scale up to considerable dimensionalities, and (2) No human and machine effort in parameter tuning on a validation set is needed, as there is a theoretically motivated "default" choice of parameter settings which have been shown to provide best results.

The key idea behind SVMs is to boost the dimension of the representation vectors and then to find the best line or hyper-plane from the widest set of parallel hyper-planes. This hyper-plane maximizes the distance between two elements in the set. The elements in the set are the support vectors. Theoretically, the classifier is determined by a very small number of examples defining the category frontier, the support vectors. Practically, finding the support vectors is not a trivial task.

Training SVMs. The implementation used is SVM-light v.5 (Joachims, 1999). The classifier was run with its default setting, with linear kernel function and no kernel optimization tricks. The SVM-light was trained on the very same (meta-tagged) training corpus the naive classifier was trained on. Since SVM is supposed to be robust and to fit big and noisy collections, no feature selection method was applied. The special feature underlying SVMs is the high dimensional representation of a document, allowing categorization by a hyper-plane of high dimension; therefore each document was represented by the vector of its stems. The dimension was boosted to include all the *stems* from the positive set. The boosted vector dimension was 7935, the number of different stems in the collection. The number of support vectors discovered was 17, which turned out to be too small for this task. Testing this model on the test set (the same test set used to test the naive classifier from previous section) yielded very poor results. It seemed that the classification was totally random. Testing the classifier on smaller subsets of the training set (200, 300, 400 documents) exposed signs of convergence, suggesting the training set is too sparse for the SVM.

To overcome sparse data, more documents were needed. The size of the training set was more than doubled. A total of 9968 documents was used as the training set. Just like the original training set, most of the biographies were randomly extracted from `biography.com`, while a few dozen biographies were manually extracted from various online sources to correct for a possible bias in `biography.com`. Training SVM-light on the new training set yielded 232 support vectors, which seems enough to perform this classification tasks.

6 Experimental Results

In order to test the effectiveness of the biography classifiers in improving question answering, we integrated each one of them with the naive baseline biographical QA system and tested the integrated system, called BioGrapher (Figure 1). Before discussing the results of this experiment, we briefly mention how the two classifiers performed on the pure classification task. For this purpose, we created a test set including 47 documents that were retrieved from the web. The evaluation measure was the accuracy of the classifiers in recognizing biographies. The Ripper-based algorithm achieved 89% success, outranking the SVM which achieved 83%. A discussion of this difference is beyond the scope of this paper (see (Tsur, 2003) for details).

We tested BioGrapher on 11 out of the 30 biographical questions in the TREC 2003 QA track. Those 11 questions were chosen as a test set because the baseline system (Section 3) scored poorly on them, suggesting that our baseline heuristics are incapable of effectively dealing with this type of questions.

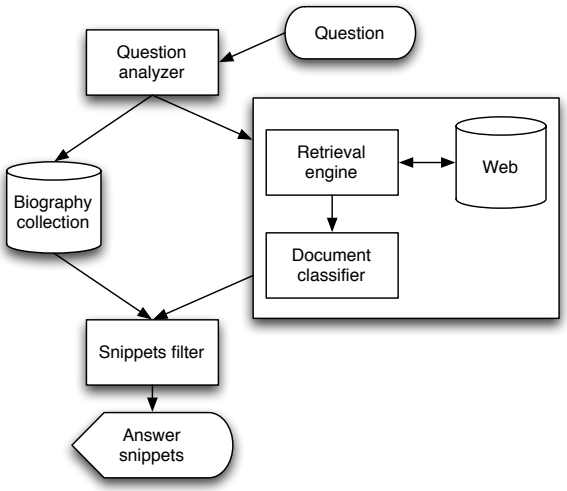

Figure 1: BioGrapher system overview

Two experiments were carried out, one for the Ripper-based classifier and another for the SVM-based one. For each definitional question Biographer submits two simple queries to a web search engine (e.g., *Sir John Hale* and *Sir John Hale biography*). It retrieves the top 20 documents returned by the search engine, thus obtaining, for each question, 40 documents amongst which it should find a biography. BioGrapher then classifies the documents into biographies and non-biographic documents. The distribution of documents that were classified as biographies can be found in Table 4.

To simplify the experiments, and especially the error analysis, we set up BioGrapher to return answer snippets from a single biography or biography-like document only. Recall, the test questions were such that there were no biographies for the question targets in the biography collection we used (`biography.com`): the biographies used were ones that BioGrapher identified on the web. We evaluated BioGrapher in the following manner. The assessor first determines whether the document from which BioGrapher extracts answer snippets is a proper biography or not. In case the document is not a pure biography the F-score given to this question is zero. Otherwise, the F-score was determined in the manner described in Section 3.[2]

Question	*Naive Classifier*	*SVM*
Who is Alberto Tomba?	2	4
Who is Albert Ghiorso?	8	2
Who is Alexander Pope?	13	11
Who is Alice Rivlin?	3	2
Who is Absalom?	2	3
Who is Nostradamus?	1	1
Who is Machiavelli?	3	6
Who is Andrea Bocceli?	1	1
Who is Al Sharpton?	2	6
Who is Aga Khan?	4	1
Who is Ben Hur?	2	1

Table 4: Distribution of documents retrieved (including false-positive)

BioGrapher with the Ripper-based Classifier

The total number of documents that were classified as biographies is 41 (out of 440 retrieved documents). However, analysis of the results reveals that the false positive ratio is high; only 20 of the 41 chosen documents were proper biography-like pages, the other "biographies" were very noisy.

For 4 out of the 11 test questions, a proper biography was returned as the top ranking document. While all 4 questions scored 0 at the original TREC evaluation, now their average F-score is 0.732, improving the average F-score over all biography questions by 9.6% to 0.4659.

BioGrapher with the SVM Classifier

The total number of documents that were classified as biographies is 38 (out of 440 retrieved docu-

[2]Obviously, the F-score for snippets extracted from documents incorrectly classified as biographies could be higher than zero because these documents could still contain valuable pieces of biographical information that would contribute to the answer's F-score. However, we decided to compute precision and recall only for snippets extracted from documents correctly classified as biographies as we think of the biography classifier as a means to identify ("on-the-fly") quality documents that could in principle be added to a biography collection.

ments). However, just like in the case of the Ripper-based classifier, an analysis of the results reveals that the false positive ratio is high; only 18 of the 38 chosen documents were biography-like.

The SVM classifier managed to return proper biographies (as top ranking documents) for 5 out of 11 questions. The average F-score for those questions is 0.674 instead of the original zero, improving the average F-score over all biography questions by 9.7% to 0.4665.

No biographies at all were retrieved for 4 of the 11 definition targets in the test set, the same four definition targets for which the Ripper-based classifier did not find biographies. A closer look reveals the same problems as with the Ripper-based classifier: a relatively high false-positive error ratio and weak ranking of the classified biographies.

7 Discussion

The results of the experiments using both classifiers are quite similar. The system integrated with the SVM-based classifier achieved a slightly higher F-score but it still falls within the standard deviation. Our experiment serves as a proof of concept for the hypothesis that using text classification methods improves the baseline QA system in a principled way.

In spite of the major improvement in the system's performance, we have found two main problems with the classifiers. First, although the classifiers managed to identify biography-like documents, they have a high false-positive ratio and too many errors in filtering out some of the non-pure-biography documents. This happens when the documents retrieved by the web search engine simply cannot be regarded as clean biographies by human assessors, although they do contain many biographic details. Second, most of the definition targets had biographies retrieved and even classified as biographies, but the biographies were ranked below other *noisy* biographical documents, therefore the best biography was not presented as a source from which to extract answer snippets. There are various obvious paths to improve over the current system: (1) Improve the classifiers by better training and other classification algorithms; (2) Enable the extraction of answers from 'noisy' biography-like documents in such a way that the gain in recall is not reversed by a loss of precision; and (3) Allow for the extraction of answer snippets from multiple biography-like documents, while avoiding to return overlapping snippets.

8 Conclusion

In this paper we have addressed the problem of biographical question answering. The main challenge in this restricted domain is the fact that the available collections of biography documents are (unavoidably) too small to admit answering all biographical questions. We use the web as a backoff source for finding biography-like documents from which to extract answers in case a given biography collection does not contain information about the question target. We demonstrated the benefits of integrating a text classifier into a restricted domain QA system as a filter to web retrieval. Finally, we believe that the use of text classifiers can be beneficial for definitional QA, especially for identifying documents from which the final answer should be extracted. Future work will address the weakness of the current implementation: improved biography classification and improved answer extraction from biography-like documents.

Acknowledgments

Maarten de Rijke was supported by the Netherlands Organization for Scientific Research (NWO) under project numbers 612-13-001, 365-20-005, 612.069.006, 612.000.106, 220-80-001, 612.000.207, and 612.066.302.

References

J. Chu-Carroll and J. Prager. 2002. Use of WordNet hypernyms for answering what-is questions. In *Proceedings of the Tenth Text REtrieval Conference (TREC 2001)*. NIST Special Publication 500-250.

W. Cohen and Y. Singer. 1996. Context sensitive learning methods. In *Proceedings of the 19th ACM International Conference on Research and Development in Information Retrieval (SIGIR-96)*, pages 307–315. ACM Press.

E. Hovy, U. Hermjakob, and C.Y. Lin. 2002. The use of external knowledge in factoid QA. In *Proceedings of the Tenth Text REtrieval Conference (TREC 2001)*. NIST Special Publication 500-250.

V. Jijkoun, G. Mishne, C. Monz, M. de Rijke, S. Schlobach, and O. Tsur. 2004. The University of Amsterdam at the TREC 2003 Question Answering Track. In E.M. Voorhees, editor, *Proceedings TREC 2003*. NIST Special Publication SP 500-255.

T. Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of ECML-98, 10th European Conference on Machine Learning*.

T. Joachims. 1999. Svm-light v.5, making large-scale svm learning practical. advances in kernel methods - support vector learning. B. Scholkopf and C. Burges and A. Smola (ed.) MIT-Press.

D.D. Lewis. 1992. *Representation and learning in information retrieval*. Ph.D. thesis, Graduate School of the University of Maassachusetts.

D.D. Lewis. 1998. Naive (bayes) at forty: The independence assumption in information retrieval. In *Proceedings of the 10th European Conference on Machine Learning*, pages 137–142. Springer-Verlag.

C.Y. Lin. 2002. The effectiveness of dictionary and web based answer reranking. In *The 19th International Conference on Computational Linguistics (COLING 2002)*.

F. Sebastiani. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47.

O. Tsur. 2003. Definitional question answering using trainable text classifiers. Master's thesis, ILLC, University of Amsterdam.

E.M. Voorhees. 2004. Overview of the TREC 2003 question answering track. In *Text REtrieval Conference (TREC 2004)*. NIST Special Publication: SP 500-255.

H. Yang, T.-S. Chua, S. Wang, and C.-K. Koh. 2003. Structured use of external knowledge for event-based open domain question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 33–40. ACM Press.