Semantic Retrieval Approach to Factoid Question Answering for Bulgarian

Stanislav Peshterliev[1] and Ivan Koychev[1,2]

[1]Faculty of Mathematics and Informatics,
University of Sofia "St. Kliment Ohridski", Sofia, Bulgaria
[2]Institute of Mathematics and Informatics at Bulgarian Academy of Sciences
{speshterliev,koychev}@fmi.uni-sofia.bg

Abstract. With the abundance of information available today, we need efficient tools to explore it. Search engines attempt to retrieve the most relevant documents for a given query, but still require users to look for the exact answer. Question Answering (Q&A) systems go one step further by trying to answer users' questions posed in natural language. In this paper we describe a semantic approach to Q&A retrieval for Bulgarian language. We investigate how the usage of named entity recognition, question answer type detection and dependency parsing can improve the retrieval of answer-bearing structures compared to the bag-of-words model. Moreover, we evaluate nine different dependency parsing algorithms for Bulgarian, and a named entity recognizer trained with data automatically extracted from Wikipedia.

Keywords: Question Answering, Information Retrieval, Semantic Annotation

1 Introduction

Most Q&A systems use a search engine to and relevant information for a given question and then process the results for answer extraction. We support the opinion that the improvement of the retrieval of answer-bearing structures is critical for the overall performance of Q&A systems. Consider the question *Who wrote Macbeth?*, a simple bag-of-word query will look like *wrote AND macbeth*, and it will match both sentences *Shakespeare wrote Macbeth* and *Macbeth wrote four poems*, as possible answer-bearing structures. In order to improve the query precision we can pre-process the sentences for named entity recognition and the question for expected answer type. Then we can use this information to formulate the query *(wrote AND macbeth) NEAR author* that will match sentences with the question's keywords close to its expected answer type *author*. Additionally, we can improve the retrieval precision by specifying that *author* is the subject and *Macbeth* is the object of the verb *wrote*.

What is the improvement to the retrieval precision that the described semantic approach brings compared to bag-of-words model? In this paper we evaluate how question answer type detection, named entity recognition and dependency parsing affects the retrieval of answer-bearing structures for Bulgarian language. For the evaluation,

we developed a test set of questions for Bulgarian that is based on the test set available for TREC 2007 Q&A Track. We also investigate the application of the available Bulgarian linguistic resources for the training of Malt dependency parser. Additionally, we present the performance of a named entity recognizer built with data extracted from Wikipedia (`http://wikipedia.org`) and DBpedia (`http://dbpedia.org`).

2 Related Work

This paper in largely influenced by the work of Mihalcea and Moldovan [3], and Bilotti et al. [1], both evaluated for English with data from TREC Q&A track (`http://trec.nist.gov`). Mihalcea and Moldovan have shown how question answer type detection and named entity recognition reduce two times the number of candidates for answer extraction. Bilotti et al. have researched how semantic role labeling can contribute to even more precise query formulation, and measured the impact of the annotation quality on Q&A retrieval. Other important Bilotti et al. contribution is that they demonstrated that scoring based on term dependency in better then proximity keyword occurrences used by Mihalcea and Moldovan. Tiedemann [11] used dependency parsing in combination with genetic based algorithm for query features selection for Dutch Q&A retrieval and answer extraction, achieving considerable difference compared to bag-of-words model.

Simov and Osenova described the architecture of BulQA [9], a Q&A system for Bulgarian language with a question analysis module, an interface module and an answer extraction module, which was evaluated at CLEF 2005. The main difference between BulQA and our work is that we focus on the retrieval for Q&A, whereas Simov and Osenova focus on the answer extraction and other problems that require sophisticated natural language processing. As a part of the work on BulQA the team also investigated the adoption of available Bulgarian language resources [6] to Q&A. Moreover, Georgiev, Nakov, Osenova, and Simov [2] evaluated the application of Maximum Entropy models for sentence splitting, tokenizing, part-of-speech tagging, chunking and synthetic parsing.

3 Retrieval for Question Answering

In this section we investigate the problems of the retrieval for Q&A for Bulgarian language. To give context for further discussions let's consider the following question and its typical bag-of-word query:

Question 1: Koj e napisal Makbet? (in English: Who wrote Macbeth?)

```
Query 1: #combine[sentence](napisa makbet)
```

Note that we are using Indri query language [10] for Query 1, where the `#combine [sentence]` clause specify that the keywords should be in one sen-

tence. Also, pay attention that the keywords are stemmed *napisal → napisa*, and question words are not included because they are part of the stop words list.

3.1 Query Formulation

How can we improve Query 1? The simplest way is to elaborate it by adding extra information. This is often done by users when they get poor results and try to add more keywords to the query to make it more precise. For Question 1 we can include *avtor (author),* which is the question's expected answer.

```
Query 2: #combine[sentence](napisa makbet avtor)
```

However, not all questions have one word or phrase answers as in the example. Questions are categorized into two groups by their expected answer type: factoid questions that have short specific answers and procedural questions, with answers several sentences or paragraphs. For example, in English the questions *When, Who* and *Where*, expect answers like time, person and location, whereas *Why* and *How* expect long answers such as tutorial or manual. The same examples are valid for Bulgarian questions: *Koga (When), Koj, Koja, Koe (Who), Kyde (Where), Zashto (Why) and Kak (How to)*. In this paper we focus on the problems of factoid question answering for Bulgarian.

3.2 Retrieval

Let's consider Query 2 that contains the expected answer. There are cases when directly injected expected answer can result in lower recall. For instance, from the following two sentences, Query 2 will match only Sentence 2 as a relevant sentence because Sentence 1 does not contain the keyword *avtor (author)*.

Sentence 1: Shekspir e napisal Makbet.
(in English: Shakespeare wrote Macbeth.)
Sentence 2: Izvesten avtor e napisal piesa za Makbet.
(in English: Famous author wrote a play for Macbeth.)

On other hand, Query 2 can match irrelevant sentences due to the lack of knowledge about predicate-arguments structures. To illustrate the problem we will use the following sentences:

Sentence 3: Izvestnijat avtor Shekspir e napisal Makbet.
(in English: The famous author Shakespeare wrote Macbeth.)
Sentence 4: Izvestnijat avtor Makbet e napisala kniga za John.
(in English: The famous author Macbeth wrote a book about John.)

Here both sentences contain the keywords from Query 2, although only Sentence 3 is relevant, because, in Sentence 4, *Makbet (Macbeth)* is the subject, but we are searching for sentences where *Makbet (Macbeth)* is the object of the verb *napisal (wrote)*.

4 Employing the Semantic Information

Here we describe how question answer type detection, named entity recognition and dependency parsing can solve the introduced difficulties in retrieving relevant sentences for answer extraction.

4.1 Named Entities and Answer Types

Named entities are words such as the names of persons, organizations, locations, expressions of times and quantities. Finding named entities in unstructured text is one of the main information extraction tasks. It is common to use ontology classes for named entity classes because you can use the ontology data to train recognition tools and take advantage of well defined ontology hierarchies as *Person → Writer*. For example if we can recognize *Writer* named entity, Sentence 1 can be annotated as follows:

Sentence 5: <Writer>Shekspir</Writer> e napisal Makbet.

Although in some cases using only named entity annotations can improve the precision of the retrieval for Q&A, they alone are not enough. We can automatically detect factoid questions expected answer type, and combine it with the named entities information to formulate more precise queries. To achieve this expected answer type classes and named entity classes should be mapped to the same ontology. We can classify question's expected answer type by its question word, for example: `Koj, Koja, Koe (Who)` → `Person, Kyde (Where)` → `Location, Koga (When)` → `Date.`

By leveraging named entity recognition and answer type detection we can change Query 2 to:

```
Query 3: #combine[sentence](napisa makbet #any:writer)
```

To see the difference between Query 2 and Query 3, let's consider the plain Sentence 2 and the annotated Sentence 5 as possible retrieval candidates. Sentence 5 is more relevant answer to Question 1 than Sentence 2 because it contains expected type of answer-bearing named entity for the given factoid question. In this case structured Query 3 will retrieve the more relevant Sentence 5 be-cause the clause `#any:writer` will match the annotation `<Writer>...<Writer>`, whereas the bag-of-words Query 2 will retrieve Sentence 2. This example illustrates that bag-of-words queries lack the necessary constraints required for Q&A.

4.2 Dependency Parsing

We have seen that Query 3 is better that Query 2, but it is still not good enough to retrieve the relevant sentence between Sentence 3 and Sentence 4. The problem is that Query 3 does not specify that the writer is the subject and *Makbet (Macbeth)* is the object of the sentence. The problem can be solved by annotating the sentences with predicate-arguments information, and then use it in the query. To do this, we use de-

pendency parsing - an approach to automatic syntactic analysis inspired by theoretical linguistics. For instance, the result from the dependency parsing for Sentence 3 will be: `mod(Izvestnijat, avtor) subj(avtor, e) mod(Shekspir, avtor) ROOT(e) comp(napisal, e) obj(Makbet, e)`, which is a tree structure with the head verb as a root, and the other sentence structures (i.e. subject, object and complements) as sub-trees. However, to make this information useful for our problem we join the dependencies into larger groups, which are then added as annotations. The predicate-argument annotations for Sentence 3 and Sentence 4 will be:

Sentence 6: <Subj>Izvestnijat avtor <Writer>Shekspir</Writer> </Subj> <Root>e napisal</Root> <Obj>Makbet</Obj>.

Sentence 7: <Subj>Izvestnijat avtor <Writer>Makbet</Writer> </Subj> <Root>e napisala</Root> <Obj>kniga</Obj> <Prepcomp>za Dzhon</Prepcomp>.

To formulate the query we are processing the question with the dependency parser, and then we are constructing the query using predicate-argument structure annotations. For example Question 1 will look like:

```
Query 4: #combine[sentence](
  #combine[./subj](#any:writer)
  #combine[./root] (napisa) #combine[./obj](makbet)
)
```

Query 4 will match only the relevant Sentence 7 that contains the answer as a subject. Using this approach we can retrieve sentences for questions like *Koga Shekspir e napisal Makbet? (in English: When did Shakespeare wrote Macbeth?).* Here the expected answer type is *Date*, which will be a prepositional complement of the possible answer-bearing sentence. So the query will be:

```
Query 5: #combine[sentence](
  #combine[./subj](shekspir) #combine[./root](napisa)
  #combine[./obj](maklbet)#combine[./prepcomp](#any:date)
)
```

5 Experiments

A similar approach to Q&A retrieval was evaluated for English by Bilotti et al. We believe that with the current state of tools and linguistic resources available for Bulgarian language, we can achieve similar results.

5.1 Information Retrieval System

For the experiments we used Galago toolkit (`http://galagosearch.org`), one of the components of Lemur project. It includes the distributed computation framework TupleFlow that manages the difficult parts of the text processing. The retrieval system

supports variant of the Indri query language that provides the necessary constrain checking for the task.

5.2 Linguistic Processing

We use OpenNLP (`http://opennlp.sourceforge.net`) tools for tokenization, sentence detection, part of speech tagging. Georgiev, Nakov, Osenova and Simov [2] performed the evaluation of the tools for Bulgarian language. For dependency parsing we use MaltParser (`http://maltparser.org`) with LIBLINEAR machine learning package, trained for Bulgarian with the data from Bultreebank project [7]. We have implemented a stemmer based on BulStem [4] Bulgarian language stemmer developed by Preslav Nakov. Stop word list is also available from Bultreebank project.

For named entity recognition, we have built a training corpus based on the data from Wikipedia and DBpedia formatted for training OpenNLP named finder component. Named entity classes are mapped to the DBpedia Ontology, which allow us to add richer annotation.

Question answer types detection is implemented with a set of hand-coded rules for Bulgarian, which rile on the question word and of other keywords.

5.3 Testing Corpus

The testing corpus is based on the Bulgarian version of Wikipedia dump from March 2011, which contains 173459 articles. The data from the dump is extracted in separated XML les which are stripped from the wiki mark-up, and enriched with annotations for paragraphs, sentence boundaries, named entities and predicate-argument structures.

We have developed a test set of 100 factoid questions, based on the test set available for TREC 2007 Q&A Track. For each question we have made manual relevance judgments in the testing corpus. The questions cover diverse topics and are divided in two types single and multiple answers.

5.4 Results

We used Galago compared evaluation tool to measure the difference between the described semantic approach and bag-of-word model. We achieved an average precision improvement of 9.3%, which is a less than that that reported for English [1] and Dutch [11]. The difference is not considerable because the testing corpus is not big and the content consists only of encyclopedic articles.

Table 1 contains the results from the evaluation of MaltParser with different dependency parsing algorithms on the data from Bultreebank project. From the available machine learning packages, we have used only LIBLINEAR because both training and parsing with LIBSVN were too slow in our experiments. For all other parameters, we kept the default values. Our results are in the range between 80%-90% reported by Nivre et al. [5] for various other languages, when no language specific optimizations

are applied. Stack lazy performed best with ac-curacy of 89.8551%, but the difference compared to the other parsing algorithms is not significant, and it may change depending on the data.

Table 1. MaltParser evaluation result

Parsing Algorithm	Accuracy
Nivre arc-eager	89.1810
Nivre arc-standard	89.2990
Covington non-projective	88.4395
Covington projective	87.8834
Stack projective	89.4169
Stack eager	89.7371
Stack lazy	***89.8551***
Planar eager	88.5912
2-Planar eager	89.4506

In Table 2 we provide evaluation for the top seven largest classes from the first level of DBpedia ontology. The low recall for persons, places and species shows that classes with entities from many different languages are hard to generalize. Another observation is that the larger one class is the harder is to detect its entities. In practice we use a combination of these models, dictionaries and regular expressions to perform named entity recognition.

Table 2. OpenNLP Name Finder evaluation results

	Sentences	Names	Precision	Recall	F_1-Measure
Place	271367	10506	0.7094	0.2347	0.3528
Person	70922	10691	0.8625	0.4337	0.5772
Organisation	20691	2826	0.8560	0.7343	0.7905
Species	19590	3176	0.7295	0.2548	0.3776
Language	16864	802	0.7475	0.6848	0.7148
EthnicGroup	14872	720	0.8628	0.7276	0.7894
Event	13521	2165	0.8390	0.8386	0.8388

6 Conclusions and Further Work

As confirmed for English and Dutch, the described approach to Q&A retrieval improves the precision and reduces the total number of retrieved documents. The reduction in the number of candidates for answer extraction mean that more sophisticated performance intensive algorithms can be used at this stage. Moreover, we confirmed that there are state-of-the-art-quality linguistic resources available for Bulgarian thanks to Kiril Simov, Petya Osenova and other contributors to Bultreebank project.

We have focused on factoid questions, but term dependencies can also be used for complex procedural Q&A retrieval where answer is several sentences or paragraphs.

Scaling question answer type detection to a large ontology is a challenging task, it will be interesting to perform experiments similar to these of Roberts and Hickl [8] with machine learning from large corpus for Bulgarian, and test how it will impact the performance of the system with more diverse and complex questions.

Acknowledgments. We are grateful to Kiril Simov and Petya Osenova for providing us with the data from Bultreebank project, and their help with references and suggestion. This research is supported by the SmartBook project, subsidized by the Bulgarian National Science Fund, under Grant D002-111/15.12.2008.

References

1. Bilotti, M.W., Ogilvie, P., Callan, J., Nyberg, E.: Structured retrieval for question answering. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 351-358. SIGIR '07, ACM, New York, NY, USA (2007)
2. Georgiev, G., Nakov, P., Osenova, P., Simov, K.: Cross-lingual adaptation as a base-line: adapting maximum entropy models to bulgarian. In: Proceedings of the Workshop on Adaptation of Language Resources and Technology to New Domains. pp. 35-38. AdaptLRTtoND '09, Association for Computational Linguistics, Stroudsburg, PA, USA (2009)
3. Mihalcea, R., Moldovan, D.I.: Document indexing using named entities Studies in Informatics and Control, Vol. 10, No. 1. (2001), pp. 21-28. (2001)
4. Nakov, P.: Building an in inflectional stemmer for Bulgarian. Proceedings of the International conference on computer systems and technologies - CompSysTech03.
5. Nivre, J., Hall, J., Nilsson, J.: Maltparser: A data-driven parser-generator for dependency parsing. In: In Proc. of LREC-2006. pp. 2216-2219 (2006)
6. Osenova, P., Simov, A., Simov, K.I., Tanev, H., Kouylekov, M.: Bulgarian-English question answering: Adaptation of language resources. In: Peters, C., Clough, P., Gonzalo, J., Jones, G.J.F., Kluck, M., Magnini, B. (eds.) CLEF. LNCS, vol. 3491, pp. 458-469. Springer (2004)
7. Osenova, P., Simov, K.: Btb-tr05: Bultreebank stylebook. bultreebank project tech-nical report n 05. (2004)
8. Roberts, K., Hickl, A.: Scaling answer type detection to large hierarchies. In: LREC. European Language Resources Association (2008)
9. Simov, K.I., Osenova, P.: BulQA: Bulgarian-bulgarian question answering at CLEF 2005. In: Peters, C., Gey, F.C., Gonzalo, J., Muller, H., Jones, G.J.F., Kluck, M., Magnini, B., de Rijke, M. (eds.) CLEF. LNCS, vol. 4022, pp. 517-526. Springer (2005)
10. Strohman, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: a language-model based search engine for complex queries. Tech. rep., in Proceedings of the International Conference on Intelligent Analysis (2005)
11. Tiedemann, J.: Improving passage retrieval in question answering using NLP. In: Bento, C., Cardoso, A., Dias, G. (eds.) EPIA. LNCS, vol. 3808, pp. 634-646. Springer (2005)