Question Analysis for Polish Question Answering

Piotr Przybyła
Institute of Computer Science, Polish Academy of Sciences,
ul. Jana Kazimierza 5, 01-248 Warszawa, Poland,
P.Przybyla@phd.ipipan.waw.pl

Abstract

This study is devoted to the problem of question analysis for a Polish question answering system. The goal of the question analysis is to determine its general structure, type of an expected answer and create a search query for finding relevant documents in a textual knowledge base. The paper contains an overview of available solutions of these problems, description of their implementation and presents an evaluation based on a set of 1137 questions from a Polish quiz TV show. The results help to understand how an environment of a Slavonic language affects the performance of methods created for English.

1 Introduction

The main motivation for building Question Answering (QA) systems is that they relieve a user of a need to translate his problem to a machine-readable form. To make it possible, we need to equip a computer system with an ability to understand requests in a natural language, find answers in a knowledge base and formulate them in the natural language. The aim of this paper is to deal with the first of these steps, i.e. **question analysis** module. It accepts the question as an input and returns a data structure containing relevant information, herein called **question model**. It consists of two elements: a **question type** and a **search query**.

The question type classifies a question to one of the categories based on its structure. A **general question type** takes one of the following values: verification (*Czy Lee Oswald zabił Johna Kennedy'ego?*, Eng. *Did Lee Oswald kill John Kennedy?*), option choosing (*Który z nich zabił Johna Kennedy'ego: Lance Oswald czy Lee Oswald?*, Eng. *Which one killed John Kennedy: Lance Oswald or Lee Oswald?*), named entity (*Kto zabił Johna Kennedy'ego?*, Eng. *Who killed John Kennedy?*), unnamed entity (*Czego użył Lee Oswald, żeby zabić Johna Kennedy'ego?*, Eng. *What did Lee Oswald use to kill John Kennedy?*), other name for a given named entity (*Jakiego pseudonimu używał John Kennedy w trakcie służby wojskowej?*, Eng. *What nickname did John Kennedy use during his military service?*) and multiple entities (*Którzy prezydenci Stanów Zjednoczonych zostali zabici w trakcie kadencji?*, Eng. *Which U.S. presidents were assassinated in office?*). There are many others possible, such as definition or explanation questions, but they require specific techniques for answer finding and remain beyond the scope of this work. For example, the *Question Answering for Machine Reading Evaluation* (QA4MRE) competition (Peñas et al., 2012) included these complex questions (e.g. *What caused X?*, *How did X happen?*, *Why did X happen?*). In case of named entity questions, it is also useful to find its **named entity type**, corresponding to a type of an entity which could be provided as an answer. A list of possible options, suited to questions about general knowledge, is given in Table 1. As some of the categories include others (e.g. CITY is a PLACE), the goal of a classifier is to find the narrowest available.

The need for a **search query** is motivated by performance reasons. A linguistic analysis applied to a source text to find the expected answer is usually resource-consuming, so it cannot be performed on the whole corpus (in case of this experiment 839,269 articles). To avoid it, we transform the question into the search query, which is subsequently used in a search engine, incorporating a full-text index of the corpus. As a result we get a list of documents, possibly related to the question. Although the query generation plays an auxiliary role, failure at this stage may lead both to too long processing times (in case of excessive number of returned documents) and lack of a final answer (in

Proceedings of the ACL Student Research Workshop, pages 96–102,
Sofia, Bulgaria, August 4-9 2013. ©2013 Association for Computational Linguistics

Question type	Occurrences
NAMED_ENTITY	657
OPTION	28
VERIFICATION	25
MULTIPLE	28
UNNAMED_ENTITY	377
OTHER_NAME	22
PLACE	33
CONTINENT	4
RIVER	11
LAKE	9
MOUNTAIN	4
RANGE	2
ISLAND	5
ARCHIPELAGO	2
SEA	2
CELESTIAL_BODY	8
COUNTRY	52
STATE	7
CITY	52
NATIONALITY	12
PERSON	260
NAME	11
SURNAME	10
BAND	6
DYNASTY	6
ORGANISATION	20
COMPANY	2
EVENT	7
TIME	2
CENTURY	9
YEAR	34
PERIOD	1
COUNT	31
QUANTITY	6
VEHICLE	10
ANIMAL	1
TITLE	38

Table 1: The 6 general question types and the 31 named entity types and numbers of their occurrences in the test set.

case of not returning a relevant document).

2 Related work

The problem of determination of the general question type is not frequent in existing QA solutions, as most of the public evaluation tasks, such as the *TREC question answering track* (Dang et al., 2007) either provide it explicitly or focus on one selected type. However, when it comes to named entity type determination, a proper classification is indispensable for finding an answer of a desired type. Some of the interrogative pronouns, such as *gdzie* (Eng. *where*) or *kiedy* (Eng. *when*) uniquely define this type, so the most obvious approach uses a list of manually defined patterns. For example, Lee et al. (2005) base solely on such rules, but need to have 1273 of them. Unfortunately, some pronouns (i.e. *jaki*, Eng. *what*, and *który*, Eng. *which*) may refer to different types of entities. In questions created with them, such as *Który znany malarz twierdził, że obciął sobie ucho?* (Eng. *Which famous painter claimed to have cut his ear?*) the **question focus** (*znany malarz*, Eng. *famous painter*), following the pronoun, should be analysed, as its type corresponds to a named entity type (a PERSON in this case). Such approach is applied in a paper by Harabagiu et al. (2001), where the *Princeton WordNet* (Fellbaum, 1998) serves as an ontology to determine foci types. Finally, one could use a machine learning (ML) approach, treating the task as a classification problem. To do that, a set of features (such as occurrences of words, beginning pronouns, etc.) should be defined and extracted from every question. Li and Roth (2002) implemented this solution, using as much as 200,000 features, and also evaluated an influence of taking into account hierarchy of class labels. Čeh and Ojsteršek (2009) used this approach in a Slovene QA system for closed domain (students' faculty-related questions) with a SVM (support vector machines) classifier.

The presented problem of question classification for Polish question answering is studied in a paper by Przybyła (2013). The type determination part presented here bases on that solution, but includes several improvements.

To find relevant documents, existing QA solutions usually employ one of the widely available general-purpose search engines, such as *Lucene*. Words of the question are interpreted as keywords and form a boolean query, where all the constituents are considered required. This procedure suffices only in case of a web-based QA, where we can rely on a high redundancy of the WWW, which makes finding a similar expression probable enough. Such an approach, using the *Google* search engine is presented by Brill et al. (2002). When working with smaller corpora, one needs to take into account different formulations of the desired information. Therefore, an initial query is subject to some modifications. First, some of the keywords may be dropped from the query; Moldovan et al. (2000) present 8 different heuristics of selecting them, based on quotation marks, parts of speech, detected named entities and other features, whereas Katz et al. (2003) drop terms in order of increasing IDF. Čeh and Ojsteršek (2009) start term removal from the end of the sentence. Apart from simplifying the query, its expansion is also possible. For example, Hovy et al. (2000) add synonyms for each keyword, extracted from *WordNet* while Katz et al. (2003) introduce their inflectional and derivational morphological forms.

3 Question analysis

For the purpose of building an open-domain corpus-based Polish question answering system, a question analysis module, based on some of the solutions presented above, has been implemented. The module accepts a single question in Polish and outputs a data structure, called a **question model**. It includes a general question type, a set of named entity types (if the general type equals NAMED_ENTITY) and a *Lucene* search query. A set of named entity types, instead of a single one, is possible as some of the question constructions are ambiguous, e.g. a *Kto?* (Eng. *Who?*) question may be answered by a PERSON, COUNTRY, BAND, etc.

3.1 Question type classification

For the question type classification all the techniques presented above are implemented. Pattern matching stage bases on a list of 176 regular expressions and sets of corresponding question types. If any of the expressions matches the question, its corresponding set of types may be immediately returned at this stage. These expressions cover only the most obvious cases and have been created using general linguistic knowledge. The length of the list arises from some of the features of Polish, typical for Slavonic languages, i.e. relatively free word order and rich nominal inflection (Przepiórkowski, 2007). For example one English pattern *Whose ... ?* corresponds to 11 Polish patterns (*Czyj ... ?*, *Czyjego ... ?*, *Czyjemu ... ?*, *Czyim ... ?*, *Czyja ... ?*,*Czyjej ... ?*, *Czyją ... ?*, *Czyje ... ?*, *Czyi ... ?*, *Czyich ... ?*, *Czyimi ... ?*).

However, in case of ambiguous interrogative pronouns, such as *jaki* (Eng. *what*) or *który* (Eng. *which*), a further analysis gets necessary to determine a question focus type. The question is annotated using the morphological analyser *Morfeusz* (Woliński, 2006), the tagger *PANTERA* (Acedański, 2010) and the shallow parser *Spejd* (Przepiórkowski, 2008). The first nominal group after the pronoun is assumed to be a question focus. The Polish WordNet database *plWordNet* (Maziarz et al., 2012) is used to find its corresponding lexeme. If nothing is found, the procedure repeats with the current group's semantic head until a single segment remains. Failure at that stage results in returning an UNNAMED_ENTITY label, whereas success leads us to a synset in WordNet. Then, we check whether its direct and indirect parents (i.e. synsets connected via hypernymy relations) include one of the predefined synsets, corresponding to the available named entity types. The whole procedure is outlined in Figure 1. The error analysis of this procedure performed in (Przybyła, 2013) shows a high number of errors caused by a lack of a word sense disambiguation. A lexeme may be connected to many synsets, each corresponding to a specific word sense and having a different parent list. Among the possible ways to combine them are: intersection (corresponding to using only the parents common for all word senses), union (the parents of any word sense), voting (the parents common for the majority of word senses) and selecting only the first word sense (which usually is the most common in the language). The experiments have shown a better precision of classification using the first word sense (84.35%) than other techniques (intersection - 72.00%, union - 80.95%, voting - 79.07%). Experimental details are provided in the next section.

As an alternative, a machine learning approach has been implemented. After annotation using the same tools, we extract the features as a set of root forms appearing in the question. Only the lemmas appearing in at least 3 sentences are used for further processing. In this way, each sentence is described with a set of boolean features (420 for the evaluation set described in next section), denoting the appearance of a particular root form. Additionally, morphological interpretations of the first five words in the question are also extracted as features. Two classifiers, implemented in the *R* statistical environment, were used: a decision tree (for human-readable results) and a random forest (for high accuracy).

3.2 Query formation

The basic procedure for creating a query treats each segment from the question (apart from the words included in a matched regular expression) as a keyword of an OR boolean query. No term weighting or stop-words removal is implemented as *Lucene* uses TF/IDF statistic, which penalizes omnipresent tokens. However, several other im-

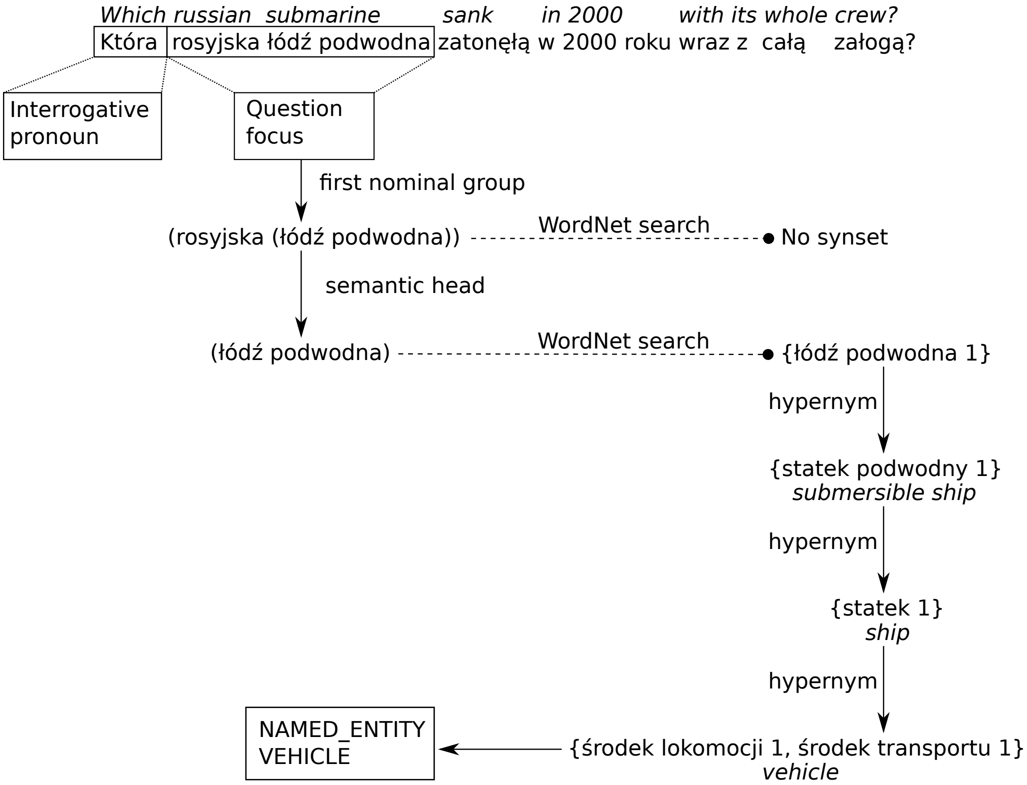

Figure 1: Outline of the disambiguation procedure, used to determine named entity type in case of ambiguous interrogative pronouns (see explanation in text).

provements are used. First, we start with a restrictive AND query and fall back into OR only in case it provides no results. A question focus removal (applied by Moldovan et al. (2000)) requires special attention. For example, let us consider again the question *Który znany malarz twierdził, że obciął sobie ucho?*. The words of the question focus *znany malarz* are not absolutely necessary in a source document, but their appearance may be a helpful clue. The query could also be expanded by replacing each keyword by a nested OR query, containing synonyms of the keyword, extracted from *plWordNet*. Both the focus removal and synonym expansion have been implemented as options of the presented query formation mechanism.

Finally, one needs to remember about an important feature of Polish, typical for a Slavonic language, namely rich nominal inflection (Przepiórkowski, 2007). It means that the orthographic forms of nouns change as they appear in different roles in a sentence. We could either ignore this fact and look for exact matches between words in the question and a document or allow some modifications. These could be done by stemming (available for Polish in *Lucene*, see the description in (Galambos, 2001)), fuzzy queries (allowing a difference between the keyword and a document word restricted by a specified Levenshtein distance) or a full morphological analysis and tagging of the source corpus and the query. All the enumerated possibilities are evaluated in this study, apart from the last one, requiring a sizeable amount of computing resources. This problem is less acute in case of English; most authors (e.g. Hovy et al. (2000)) use simple (such as Porter's) stemmers or do not address the problem at all.

4 Evaluation

For the purpose of evaluation, a set of 1137 questions from a Polish quiz TV show *"Jeden z dziesięciu"*, published in (Karzewski, 1997), has been manually reviewed and updated. A general question type and a named entity type has been assigned to each of the questions. Table 1 presents the number of question types occurrences in the test set. As a source corpus, a textual version of the Polish Wikipedia has been used. To evaluate query generation an article name has been assigned to those questions (1057), for which a single article in Wikipedia containing an answer exists.

Outputs of type classifiers have been gathered

Classifier	Classified	Precision	Overall
pattern matching	36.15%	95.37%	34.48%
WordNet-aided	98.33%	84.35%	**82.94%**
decision tree	100%	67.02%	67.02%
random forest	100%	72.91%	72.91%

Table 2: Accuracy of the four question type classifiers: numbers of questions classified, percentages of correct answers and products of these two.

and compared to the expected ones. The machine learning classifiers have been evaluated using 100-fold cross-validation[1].

Four of the presented improvements of query generation tested here include: basic OR query, AND query with fallback to OR, focus segments removal and expansion with synonyms. For each of those, three types of segment matching strategies have been applied: exact, stemming-based and fuzzy. The recorded results include recall (percentage of result lists including the desired article among the first 100) and average position of the article in the list.

5 Results

The result of evaluation of classifiers is presented in Table 2. The pattern matching stage behaves as expected: accepts only a small part of questions, but yields a high precision. The WordNet-aided focus analysis is able to handle almost all questions with an acceptable precision. Unfortunately, the accuracy of ML classifiers is not satisfactory, which could be easily explained using Table 1: there are many categories represented by very few cases. An expansion of training set or dropping the least frequent categories (depending on a particular application) is necessary for better classification.

Match Query	Exact	Stemming	Fuzzy
basic OR query	69.97% 14.32	80.08% 12.90	**82.19%** **12.36**
priority for AND query	57.94% 11.36	57.07% 8.80	34.84% 7.07
with focus segments removed	62.75% 14.65	71.99% 14.00	73.34% 12.84
with synonyms	47.06% 21.42	65.64% 15.47	58.71% 16.00

Table 3: Results of the four considered query generation techniques, each with the three types of matching strategy. For each combination a recall (measured by the presence of a given source document in the first 100 returned) and an average position on the ranked list is given.

Results of considered query generation techniques are shown in Table 3. It turns out that the basic technique generally yields the best result. Starting with an AND query and using OR only in case of a failure leads to an improvement of the expected article ranking position but the recall ratio drops significantly, which means that quite often the results of a restrictive query do not include the relevant article. The removal of the question focus from the list of keywords also has a negative impact on performance. The most surprising results are those of expanding a query with synonyms - the number of matching articles grows abruptly and *Lucene* ranking mechanism does not lead to satisfying selection of the best 100. One needs to remember that only one article has been selected for each test question, whereas probably there are many relevant Wikipedia entries in most cases. Unfortunately, finding all of them manually would require a massive amount of time.

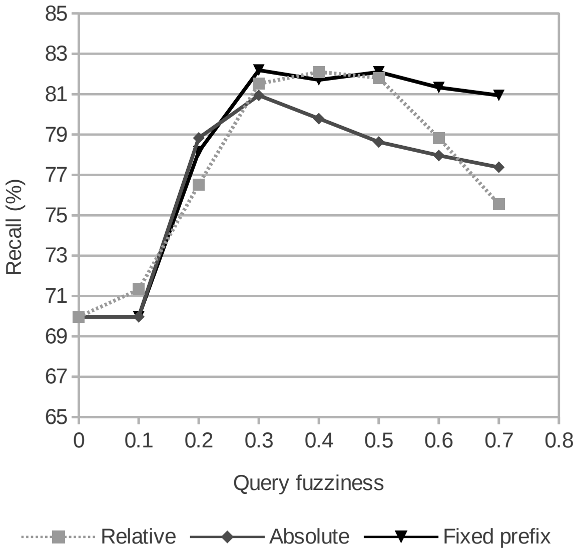

Figure 2: Impact of the fuzziness of queries on the recall using three types of fuzzy queries. To show the relative and absolute fuzziness on one plot, a word-length of 10 letters is assumed. See a description in text.

We can also notice a questionable impact of the stemming. As expected, taking into account inflection is necessary (cf. results of exact matching), but fuzzy queries provide more accurate re-

[1] I.e. the whole test set has been divided into 100 nearly equal subsets and each of them has been classified using the classifier trained on the remaining 99 subsets.

sults, although they use no linguistic knowledge.

As the fuzzy queries yield the best results, an additional experiment becomes necessary to find an optimal fuzziness, i.e. a maximal Levenshtein distance between the matched words. This parameter needs tuning for particular language of implementation (in this case Polish) as it reflects a mutability of its words, caused by inflection and derivation. Three strategies for specifying the distance have been used: relative (with distance being a fraction of a keyword's length), absolute (the same distance for all keywords) and with prefix (same as absolute, but with changes limited to the end of a keyword; with fixed prefix). In Figure 2 the results are shown - it seems that allowing 3 changes at the end of the keyword is enough. This option reflects the Polish inflection schemes and is also very fast thanks to the fixedness of the prefix.

6 Conclusion

In this paper a set of techniques used to build a question model has been presented. They have been implemented as a question analysis module for the Polish question answering task. Several experiments using Polish questions and knowledge base have been performed to evaluate their performance in the environment of the Slavonic language. They have led to the following conclusions: firstly, the best technique to find a correct question type is to combine pattern matching with the WordNet-aided focus analysis. Secondly, it does not suffice to process the first 100 article, returned by the search engine using the default ranking procedure, as they may not contain desired information. Thirdly, the stemmer of Polish provided by the *Lucene* is not reliable enough - probably it would be best to include a full morphological analysis and tagging process in the document indexing process.

This study is part of an effort to build an open-domain corpus-based question answering system for Polish. The obvious next step is to create a sentence similarity measure to select the best answer in the source document. There exist a variety of techniques for that purpose, but their performance in case of Polish needs to be carefully examined.

Acknowledgements

Critical reading of the manuscript by Agnieszka Mykowiecka is gratefully acknowledged. Study was supported by research fellowship within "Information technologies: research and their interdisciplinary applications" agreement number POKL.04.01.01-00-051/10-00.

References

Szymon Acedański. 2010. A morphosyntactic Brill Tagger for inflectional languages. In *Proceedings of the 7th international conference on Advances in Natural Language Processing (IceTAL'10)*, pages 3–14.

Eric Brill, Susan Dumais, and Michele Banko. 2002. An analysis of the AskMSR question-answering system. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - EMNLP '02*, volume 10, pages 257–264, Morristown, NJ, USA, July. Association for Computational Linguistics.

Hoa Trang Dang, Diane Kelly, and Jimmy Lin. 2007. Overview of the TREC 2007 Question Answering track. In *Proceedings of The Sixteenth Text REtrieval Conference, TREC 2007.*

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press.

Leo Galambos. 2001. Lemmatizer for Document Information Retrieval Systems in JAVA. In *Proceedings of the 28th Conference on Current Trends in Theory and Practice of Informatics (SOFSEM 2001)*, pages 243–252.

Sanda Harabagiu, Dan Moldovan, Marius Pasca, Rada Mihalcea, Mihai Surdeanu, Razvan Bunescu, Roxana Gîrju, Vasile Rus, and Paul Morarescu. 2001. The role of lexico-semantic feedback in open-domain textual question-answering. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics - ACL '01*, pages 282–289.

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk, and Chin-Yew Lin. 2000. Question Answering in Webclopedia. In *Proceedings of The Ninth Text REtrieval Conference (TREC 2000).*

Marek Karzewski. 1997. *Jeden z dziesięciu - pytania i odpowiedzi*. Muza SA.

Boris Katz, Jimmy Lin, Daniel Loreto, Wesley Hildebrandt, Matthew Bilotti, Sue Felshin, Aaron Fernandes, Gregory Marton, and Federico Mora. 2003. Integrating Web-based and corpus-based techniques for question answering. In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003).*

Changki Lee, Ji-Hyun Wang, Hyeon-Jin Kim, and Myung-Gil Jang. 2005. Extracting Template for Knowledge-based Question-Answering Using Conditional Random Fields. In *Proceedings of the 28th Annual International ACM SIGIR Workshop on MFIR*, pages 428–434.

Xin Li and Dan Roth. 2002. Learning Question Classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING-2002)*, volume 1 of *COLING '02*.

Marek Maziarz, Maciej Piasecki, and Stanisław Szpakowicz. 2012. Approaching plWordNet 2.0. In *Proceedings of the 6th Global Wordnet Conference*.

Dan Moldovan, Sanda Harabagiu, Marius Paşca, Rada Mihalcea, Roxana Gîrju, Richard Goodrum, and Vasile Rus. 2000. The structure and performance of an open-domain question answering system. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics - ACL '00*, pages 563–570, Morristown, NJ, USA, October. Association for Computational Linguistics.

Anselmo Peñas, Eduard H. Hovy, Pamela Forner, Álvaro Rodrigo, Richard F. E. Sutcliffe, Caroline Sporleder, Corina Forascu, Yassine Benajiba, and Petya Osenova. 2012. QA4MRE: Question Answering for Machine Reading Evaluation at CLEF 2012. In *CLEF 2012 Evaluation Labs and Workshop Online Working Notes*.

Adam Przepiórkowski. 2007. Slavonic information extraction and partial parsing. In *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing Information Extraction and Enabling Technologies - ACL '07*.

Adam Przepiórkowski. 2008. *Powierzchniowe przetwarzanie języka polskiego*. Akademicka Oficyna Wydawnicza EXIT, Warszawa.

Piotr Przybyła. 2013. Question classification for Polish question answering. In *Proceedings of the 20th International Conference of Language Processing and Intelligent Information Systems (LP&IIS 2013)*.

Ines Čeh and Milan Ojsteršek. 2009. Developing a question answering system for the slovene language. *WSEAS Transactions on Information Science and Applications*, 6(9):1533–1543.

Marcin Woliński. 2006. Morfeusz — a Practical Tool for the Morphological Analysis of Polish. In Mieczysław Kłopotek, Sławomir Wierzchoń, and Krzysztof Trojanowski, editors, *Intelligent Information Processing and Web Mining*, pages 511–520.