Originally published in: Hitzler, Pascal/Kirrane, Sabrina/Hartig, Olaf [et al.] (Eds.): The Semantic Web: ESWC 2019 Satellite Events. ESWC 2019 Satellite Events, Portorož, Slovenia, June 2–6, 2019, Revised Selected Papers. - Cham: Springer, 2019. Pp. 199-208. (Lecture Notes in Computer Science 11762)
DOI: *https://doi.org/10.1007/978-3-030-32327-1_39*

POSTPRINT

Stefan Falke
Leibniz-Institut für Deutsche Sprache, Mannheim, Germany; falke@ids-mannheim.de

Developing a Knowledge Graph for a Question Answering System to Answer Natural Language Questions on German Grammar

Abstract: Question Answering Systems for retrieving information from Knowledge Graphs (KG) have become a major area of interest in recent years. Current systems search for words and entities but cannot search for grammatical phenomena. The purpose of this paper is to present our research on developing a QA System that answers natural language questions about German grammar.
Our goal is to build a KG which contains facts and rules about German grammar, and is also able to answer specific questions about a concrete grammatical issue. An overview of the current research in the topic of QA systems and ontology design is given and we show how we plan to construct the KG by integrating the data in the grammatical information system *Grammis*, hosted by the Leibniz-Institut für Deutsche Sprache (IDS). In this paper, we describe the construction of the initial KG, sketch our resulting graph, and demonstrate the effectiveness of such an approach. A grammar correction component will be part of a later stage. The paper concludes with the potential areas for future research.

Keywords: Knowledge Graph, Ontology development, German Grammar, Question Answering System

1 Introduction

Questions about german grammar reach the Leibniz-Institut für Deutsche Sprache (IDS)[1] by e-mail every day. The answers are mostly more than just a *Yes* or *No*, but rather give some more information about the case, sometimes with empirical data from a corpus analysis or the latest research results as referenced papers.

For the department of grammar, the online information system *Grammis*[2] is an important source of information to answer these questions. *Grammis* is the online information system on German grammar of the IDS with more than

[1] Leibniz-Institute for the German Language.
[2] http://grammis.ids-mannheim.de [25].

Publikationsserver des Leibniz-Instituts für Deutsche Sprache
URN: http://nbn-resolving.de/urn:nbn:de:bsz:mh39-95083

3,000 descriptive texts and about 2,000 dictionary entries [25]. The information in *Grammis* is taken mostly from the three-volumes book "Grammatik der Deutschen Sprache (GDS)"[3] [30] written by the IDS. Even though most information to answer these questions is in *Grammis*, the user cannot find it or cannot apply it to his concrete case. The *user* is usually a professional writer (author, journalist), teacher or student (school teacher, german as a foreign language, linguistic student) or professional linguist.

To find information in *Grammis*, the user can either use the navigation menus or a full-text search. Both have their limitations, because to use the navigation menu the user should know in which grammatical field his question is located, e.g., to find information on the word class noun, one had to follow the navigation path: *Forschung* → *Systematische Grammatik* → *Ausdruckskategorien und Ausdrucksformen* → *Wortarten* → *Nomen*[4]. This path leads to a descriptive text about nouns. One could also use the full-text search, in which the user enters a few keywords, in this case *Nomen* (noun). The result is a ranked result list with more than 100 documents.

Both ways lead to documents, that only provide general rules on German grammar. It can be frustrating or confusing for the user to apply these rules to his specific case. It would improve the users' experience if the user could ask the system a question in his own words (i.e. in natural language) and *Grammis* would provide a concrete answer. Therefore *Grammis* should be reconstructed to handle natural language questions about grammatical problems and retrieve a specific answer from the data already available in *Grammis*.

In this paper, we describe the development of a Question Answering (QA) system that allows the user to ask a question about German grammar in natural language and to receive a specific answer to his question. To build a QA system that can answer those questions, we plan to construct a knowledge graph (KG) that is able to represent grammatical information about the German language. The KG should be able to represent data from different sources within the IDS, so that the system can retrieve the information for the answers out of the KG.

The remainder of this paper is organized as follows: In the following section we describe state of the art QA systems, ontology design, and automated grammar correction. In Sect. 3 we state the problem and the research question. Our research methodology is described in Sect. 4. Section 5 contains an evaluation of our KG. Preliminary results are presented in Sect. 6. The paper closes with a summary of our findings and potential areas of future research.

2 State of the Art

In this section, we define the terminology to distinguish *knowledge graph* and *ontology*. This is followed by a discussion of QA systems in the Semantic Web and methodologies for ontology design.

[3] Grammar of the German Language.

[4] *Research* → *Systematic Grammar* → *Categories and Forms of Expressions* → *Word Classes* → *Nouns*.

2.1 Terminology

This paper follows Paulheim's [22] definition of *knowledge graph* (for other definitions of knowledge graph, cf. [8]). According to Paulheim a KG consists of the schema (T-box) and the actual instances (A-box), but with a clear focus on the instances. We use the term *ontology* to refer to the schema of a knowledge graph (T-box) and *knowledge graph* to refer to the actual graph filled with instances (A-box) in which the schema is included, but only plays a minor role.

2.2 Question Answering in the Semantic Web

A QA system receives a question in natural language and provides the user with a concrete answer to their question rather than just a list of ranked documents [12]. In recent years, the research on QA systems querying the semantic web has grown rapidly [13,15,27]. With an increasing amount of knowledge as linked data, the need to access this data has also grown. Semantic Question Answering (SQA) bridges the gap between semantic data and the end user [13]. Höffner et.al. [13] surveyed 62 systems using linked data as data source, and identified and described seven challenges that such systems face. Two of these challenges are complex queries and the type of questions. While factual or yes-no questions directly conform to SPARQL, more complex queries are still not solved [13]. They also mention the system HAWK [28], which is a hybrid source system that retrieves the answers from both linked data and textual representation. Current QA systems retrieve the information following semantic entities, but they do not search for information following grammatical phenomena [12].

Using semantic data as data source makes the QA system independent from the underlying ontology, meaning that it is possible to extend the KG with additional information over time and that it is possible to handle unknown vocabulary in the query [15,27,31]. Ambiguities are also challenging but they are manageable, since there are a variety of disambiguation methods that have already been established [13].

Some ontologies are specially designed for linguistic purposes, like the Lemon Model [19] and LexInfo [4]. The Lemon Model is an important model for creating KGs on linguistic knowledge. Lemon represents lexical information in context with concepts and terms [19].

LexInfo provides classes and properties for linguistic cases like word classes (noun, verb, etc.) or relations (hyperonym, hyponym, etc.) [4].

2.3 Ontology Design

The design of ontologies should follow a methodology in order to prevent chaotic constructions and low quality ontologies [6]. There are several frameworks for designing ontologies [9,10,29].

We will look at the methodology by Grüninger and Fox [10] in more detail. It consists of six phases: (1) First, the Motivation Scenarios should be outlined which gave the impulse for developing the ontology. (2) With a set of Informal

Competency Questions, which are questions in natural language, the scope of the ontology should be determined. (3) The First-Order Logic Terminology will be extracted with these questions. (4) After this a set of Formal Competency Questions, which are questions in a query language, will be created. (5) With these questions the First-Order logic Axioms will be defined. (6) At last the ontology will be evaluated by using Completeness Theorems which are conditions under which the solutions to the questions are complete.

2.4 Automated Grammar Correction

Automated Grammar Correction exists for a variety of languages [2,26]. There are different approaches to grammar correction: rule-based, statistical, and syntactical [16]. Syntax-based approaches are language dependent and need much handwork till they can predict a sentence as correct or incorrect. Although rule-based techniques require many handwritten rules, this approach is language dependent. Statistic-based techniques are also language independent, but they require a large corpus to be trained on, and the test and training set need to be similar to provide good results. While the first two approaches provide error messages with the found error, the statistic-based approach does not provide an error message [16]. Since each correction system focuses on one or a handful of error classes, none of the systems is able to detect all possible error classes [2,26].

Only a handful of Grammar Correction systems exist for German grammar [7,21,24], and all of the existing Grammar Correction systems solely check a given text for grammatical errors. The Language Tool [21] is rule-based, so only implemented rules are applied and a short explanation on the found error is given. But none of these tools is embedded in a QA system and none provides deeper explanations on the found error.

3 Problem Statement and Contributions

The overall task for developing the QA system, described in this paper, is to answer concrete questions about specific grammatical phenomena. The IDS has a database of the questions that users send to the IDS and the corresponding answers that the users receive. There are currently about 50,000 entries. A subset of 500 of those questions has been manually categorized [23] into one of the five categories listed in Table 1.

About three quarters of the questions that the IDS receives fall into question categories QC1 and QC5 (see Table 1). These questions could potentially be handled automatically, but only if the factual questions (question categories QC2, QC3, and QC4) are answered first.

While current QA and Information Retrieval (IR) Systems search for the entered words or entities, respectively [12,17], our QA System will be searching for grammatical phenomena. For example, if one search string is *meiner Schwester Auto* (my sister's car), the user presumably wants to know something about the genitive, not about family relations (indicated by *Schwester* (sister)) or cars (indicated by *Auto* (car)). So it would not be helpful to the user, to

Table 1. Question categories [23]

Category	Title (with examples[a])	Ratio
QC1	Correction question / Alternative question Example: Ist *a* richtig? (engl. Is *a* correct?)	52%
QC2	Grammatical category Example: Wie ist der Plural von *a*? (engl. What is the plural of *a*?)	4%
QC3	Grammatical definition / Rule question Example: Wie lautet die Regel für *C*? (engl. What is the rule for *C*?)	9%
QC4	Lexical question example: Wie lautet das Synonym für *a*? (engl. What is a synonym for *a*?)	11%
QC5	Punctuation example: Ist das Komma im folgenden Satz richtig *a*? (engl. Is the comma in the following sentence correct *a*?)	24%

[a] represents a language object (word, phrase, term), *C* a grammatical category (Tense, Case).

present any page on which one of those words appear, because there could be pages where these words appear in another context. Instead the QA System should provide the user with information about the genitive, maybe other ways to express the given sentence, like *das Auto meiner Schwester* (the car of my sister). The user could also be presented with information about the genitive, since *Grammis* contains a genitives database, which could present more background information on the genitive use of the German language.

Therefore a KG will be used to organize the data. One advantage of semantic web technologies is that one can include different resources [1]. Since answering questions about German grammar requires a large amount of data, integrating the different resources in one KG connects the data from different sources and for different aspects. These resources can either be from within the IDS but might also be external resources like GermaNet [11].

Since there are many research projects at the IDS, their collected data could be integrated into the KG as well, to find better answers or to enrich the answers with recent research. This should be considered when designing the ontology.

We derived the following main research question from our described problem.

RQ1. How does the knowledge graph need to be built to support a Question Answering System for questions about German grammar?

In addition to the main question, there are several secondary research questions, which should also be considered during the implementation of the system.

SRQ1. What entities and relations does the ontology need to consist of?
SRQ2. Can existing QA system frameworks be extended to answer questions about German grammar?

SRQ3. Could additional (external) resources be used to improve the question answering?

The goal is to construct a QA system that is able to process the different question types and present an answer to them. The system's main purpose is to measure the effectiveness of the question answering and the proposed techniques and processes.

4 Research Methodology and Approach

In this section we discuss our construction of the QA system, and our design for the ontology of the KG.

4.1 Question Answering System

The development of the QA system will follow the principles of agile software development [5]. Starting with a prototype to explore the KG, more features will be added over time. The first version will only be able to navigate through the KG. In later versions, the question and answer components will be added. The advantages of prototyping are that at an early stage a running software is available, so that basic functionality can be tested [3]. The QA system will have a modular structure, so every component can be developed and maintained independently from the other modules.

There are some frameworks, that can serve as a starting point [13], e.g., openQA [18]. The focus on developing the QA system is on the question and data component. Different approaches exist for answer retrieval, e.g., graph exploration or machine learning approaches [27]. These should be taken into account since answering questions about grammar is a complex task. The system should also be able to find the answer in linked and textual data, c.f. the HAWK QA system [28].

4.2 Ontology Design

The ontology is designed following the methodology of Grüninger and Fox [10]. The Informal Competency Questions are based on the IDS questions database [23].

For every question category (see Table 1), the required classes and relations are defined. For example, a question out of category QC2 that asks for the plural of a word would make it necessary to have the information about the plural form in the ontology. Therefore a relation could be formulated like this: $plural(L, P)$, where l is a class for a lexical entry $l \in L$, p is the plural form of l ($p \in P$) and l and p are connected via the relation $plural(l, p)$.

The ontology will not be limited by the data that is currently available in *Grammis*. Even if entities and relations are needed that are not extractable from *Grammis*, the ontology will contain these entities. If there are existing classes and relations in other ontologies, these will be included by importing and using these ontologies, e.g., Lemon [20] and LexInfo [4].

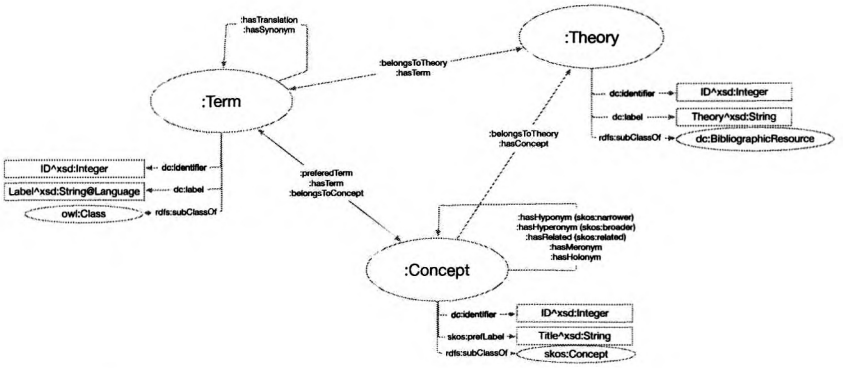

Fig. 1. Terminological classes and relations of the Ontology

5 Evaluation Plan

The evaluation will focus on achieving high precision values, while recall values are less important. This is because in order to extract the answer from the found documents, it is more important to get many true positives, than to retrieve all relevant documents but also some non-relevant documents in the result set.

Performance and scalability will be measured using the query time for a set of predefined queries. Also the query time will be measured when filling the KG with more and more facts, to compare the performance of these queries with a growing number of facts (e.g. 50 k, 100 k, 150 k facts).

Depending on the category, the questions can either be answered qualitatively or quantitatively. Since the categories QC2, QC3, and QC4 are mostly factual questions, the correct answer can be extracted from *Grammis*. These answers can be compared to the answers given by the QA system and then be categorized as *correct*, *false*, and *not answered*. In case of qualitative answers, like categories QC1 and QC5, the question database [23] works as a base for the gold standard. A set of questions and answers out of the given categories will be revised by human experts. The answers of the QA system are then categorized manually as *correct*, *incomplete*, *false*, or *not answered*.

At different stages of the development of the application, different types of questions will be possible to answer, so that in an early stage only questions from category QC2 can be answered, since these are factual questions. As the project progresses more and more questions out of the different categories should be answered correctly.

6 Preliminary Results

A web application using Flask[5] has been developed, which loads the prototype KG and lets the user navigate through it. So far, the dictionary of grammatical

[5] Flask – A Python Microframework, http://flask.pocoo.org.

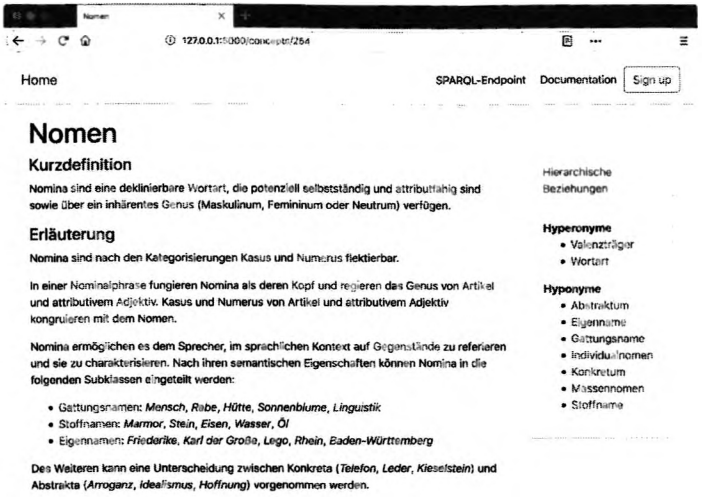

Fig. 2. Front end of Flask Application showing the entry *Nomen* (noun)

terms[6] and the orthographical dictionary[7] have been transferred into RDF and build a KG with 48,924 triples. Figure 1 shows the classes and relations of the grammatical terminology. A *concept* is a representation of an entity, while a *term* is an actual name for it. Every concept can have multiple terms. In a *theory*, a whole representation of several concepts together with their relations is defined, as well as which terms are assigned to each concept [14].

It is possible to navigate through the hierarchical relations from one concept to the other. Figure 2 shows the front end of the application. Title and description of the concept *Nomen* (noun) are in the center of the screen, on the right side there are listed the hyperonyms and hyponyms to this concept. The linked terms lead to the entry of the selected concept.

The application has a SPARQL endpoint, with which one can validate freely entered SPARQL queries against the KG. Thus it is possible to retrieve information from the KG, which later will be used to answer the factual questions. There is currently no interface to actually ask a question in natural language.

7 Conclusions

Our goal is to build a KG that is able to answer natural language questions on German grammar. Current QA systems search information by semantical entities rather than by grammatical phenomena. In addition, most grammatical

[6] Wissenschaftliche Terminologie, https://grammis.ids-mannheim.de/terminologie.
[7] Datenbank Rechtschreibwortschatz, https://grammis.ids-mannheim.de/rechtschreibwortschatz.

correction tools have been developed for English, with only a handful of German examples, all of which are not integrated into a QA system. The approach describe in this paper contributes to filling these gaps.

The main challenge will be to answer high level questions like in the categories QC1 and QC5. Currently, the system only has a web front end with which one can navigate through the graph, and a SPARQL endpoint. The prototype can currently retrieve answers to a few low level questions, but not react to actual questions, since the question component has not yet been implemented.

The grammar correction will be part of the later stage of this project, since answering factual questions is a fundamental prerequisite for grammar correction. Also, the extension to a multilingual system is taken into account. Although the system is designed for german grammar, every module will be checked for its language dependence. Every module which is language independent can be reused, while the other modules need to be adjusted.

Acknowledgments. I would like to thank Prof. Heiko Paulheim for his valuable feedback and support in the realization of this work.

References

1. Berners-Lee, T., Hendler, J., Lassila, O., et al.: The semantic web. Sci. Am. **284**(5), 28–37 (2001)
2. Bhirud, N., Bhavsar, R., Pawar, B.: Grammar checkers for natural languages: a review. Int. J. Nat. Lang. Comput. **6**(4), 1–13 (2017)
3. Bischofberger, W.R., Pomberger, G.: Prototyping-Oriented Software Development: Concepts and Tools. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-84760-8
4. Cimiano, P., Buitelaar, P., McCrae, J., Sintek, M.: Lexinfo: a declarative model for the lexicon-ontology interface. Web Seman. Sci. Serv. Agents World Wide Web **9**(1), 29–51 (2011)
5. Cockburn, A.: Agile Software Development, vol. 177. Addison-Wesley, Boston (2002)
6. Corcho, O., Fernández-López, M., Gómez-Pérez, A.: Ontological engineering: what are ontologies and how can we build them? In: Semantic web services: Theory, tools and applications, pp. 44–70. IGI Global (2007)
7. Crysmann, B., Bertomeu, N., Adolphs, P., Flickinger, D., Klüwer, T.: Hybrid processing for grammar and style checking. In: Proceedings of the 22nd International Conference on Computational Linguistics, vol. 1, pp. 153–160. Association for Computational Linguistics (2008)
8. Ehrlinger, L., Wöß, W.: Towards a definition of knowledge graphs. In: SEMANTiCS (Posters, Demos, SuCCESS) (2016)
9. Fernández-López, M., Gómez-Pérez, A., Juristo, N.: Methontology: from ontological art towards ontological engineering (1997)
10. Grüninger, M., Fox, M.S.: Methodology for the design and evaluation of ontologies (1995)
11. Hamp, B., Feldweg, H.: Germanet-a lexical-semantic net for German. In: Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications (1997)

12. Hirschman, L., Gaizauskas, R.: Natural language question answering: the view from here. Nat. Lang. Eng. **7**(4), 275–300 (2001)
13. Höffner, K., Walter, S., Marx, E., Usbeck, R., Lehmann, J., Ngonga Ngomo, A.C.: Survey on challenges of question answering in the semantic web. Seman. Web **8**(6), 895–920 (2017)
14. Lang, C., Schwinn, H., Suchowolec, K.: Grammatische Terminologie am IDS-ein terminologisches Online-Wörterbuch als ein vernetztes Begriffssystem. Sprachreport (2018)
15. Lopez, V., Uren, V., Sabou, M., Motta, E.: Is question answering fit for the semantic web?: a survey. Seman. Web **2**(2), 125–155 (2011)
16. Manchanda, B., Athavale, V.A., Kumar Sharma, S.: Various techniques used for grammar checking. Int. J. Comput. Appl. Inf. Technol. **9**(1), 177 (2016)
17. Manning, C.D., Raghavan, P., Schütze, H., et al.: Introduction to Information Retrieval, vol. 1. Cambridge University Press, Cambridge (2008)
18. Marx, E., Usbeck, R., Ngomo, A.C.N., Höffner, K., Lehmann, J., Auer, S.: Towards an open question answering architecture. In: Proceedings of the 10th International Conference on Semantic Systems, pp. 57–60. ACM (2014)
19. McCrae, J., et al.: Interchanging lexical resources on the semantic web. Lang. Resour. Eval. **46**(4), 701–719 (2012)
20. McCrae, J., Spohr, D., Cimiano, P.: Linking lexical resources and ontologies on the semantic web with lemon. In: Antoniou, G., et al. (eds.) ESWC 2011. LNCS, vol. 6643, pp. 245–259. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21034-1_17
21. Naber, D.: A rule-based style and grammar checker (2003)
22. Paulheim, H.: Knowledge graph refinement: a survey of approaches and evaluation methods. Seman. web **8**(3), 489–508 (2017)
23. Ripp, S., Falke, S.: Questions categories on German grammar information systems and their operability, article in preperation
24. Schmidt-Wigger, A.: Grammar and style checking for German. In: Proceedings of CLAW, vol. 98, pp. 76–86. Citeseer (1998)
25. Schneider, R., Schwinn, H.: Hypertext, Wissensnetz und Datenbank: die Webinformationssysteme Grammis und ProGr@mm, pp. 337–346. Ansichten und Einsichten, Institut für Deutsche Sprache, Mannheim (2014)
26. Sharma, S.K.: Rule based grammar checking systems. Int. J. Comput. Appl. Inf. Technol. **10**(1), 217–220 (2016)
27. Unger, C., Freitas, A., Cimiano, P.: An introduction to question answering over linked data. In: Koubarakis, M., et al. (eds.) Reasoning Web 2014. LNCS, vol. 8714, pp. 100–140. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10587-1_2
28. Usbeck, R., Ngomo, A.-C.N., Bühmann, L., Unger, C.: HAWK – Hybrid question answering using linked data. In: Gandon, F., Sabou, M., Sack, H., d'Amato, C., Cudré-Mauroux, P., Zimmermann, A. (eds.) ESWC 2015. LNCS, vol. 9088, pp. 353–368. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18818-8_22
29. Uschold, M., King, M.: Towards a methodology for building ontologies. In: Workshop on Basic Ontological Issues in Knowledge Sharing (IJCAI) (1995)
30. Zifonun, G., Hoffmann, L., Strecker, B., Ballweg, J.: Grammatik der deutschen Sprache, vol. 1. Walter de Gruyter, Berlin (1997)
31. Zou, L., Huang, R., Wang, H., Yu, J.X., He, W., Zhao, D.: Natural language question answering over rdf: a graph data driven approach. In: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, pp. 313–324. ACM (2014)