Methods for Efficient Ontology Lexicalization for Non-Indo-European Languages

The Case of Japanese

Bettina Lanser

A thesis presented for the degree of
Doctor rerum naturalium (Dr. rer. nat.)

Bielefeld University
June 2017

This work was supported by the Cluster of Excellence Cognitive Interaction Technology CITEC (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG).

Printed on non-ageing paper — ISO 9706

ABSTRACT

In order to make the growing amount of conceptual knowledge available through ontologies and datasets accessible to humans, NLP applications need access to information on how this knowledge can be verbalized in natural language. One way to provide this kind of information are ontology lexicons, which apart from the actual verbalizations in a given target language can provide further, rich linguistic information about them. Compiling such lexicons manually is a very time-consuming task and requires expertise both in Semantic Web technologies and lexicon engineering, as well as a very good knowledge of the target language at hand. In this thesis we present two alternative approaches to generating ontology lexicons by means of crowdsourcing on the one hand and through the framework M-ATOLL on the other hand. So far, M-ATOLL has been used with a number of Indo-European languages that share a large set of common characteristics. Therefore, another focus of this work will be the generation of ontology lexicons specifically for Non-Indo-European languages. In order to explore these two topics, we use both approaches to generate Japanese ontology lexicons for the DBpedia ontology: First, we use CrowdFlower to generate a small Japanese ontology lexicon for ten exemplary ontology elements according to a two-stage workflow, the main underlying idea of which is to turn the task of generating lexicon entries into a translation task; the starting point of this translation task is a manually created English lexicon for DBpedia. Next, we adapt M-ATOLL's corpus-based approach to being used with Japanese, and use the adapted system to generate two lexicons for five example properties, respectively. Aspects of the DBpedia system that require modifications for being used with Japanese include the dependency patterns employed by M-ATOLL to extract candidate verbalizations from corpus data, and the templates used to generate the actual lexicon entries. Comparison of the lexicons generated by both approaches to manually created gold standards shows that both approaches are viable options for the generation of ontology lexicons also for Non-Indo-European languages.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ADN	adnominalizer
CAUSE	causative
CL	classifier
COM	comitative
COP	copula
DOBJ	direct object
HON	honorative
IOBJ	indirect object
PASS	passive
PAST	past tense
POSS	possession marker
SUBJ	subject
TOP	topic

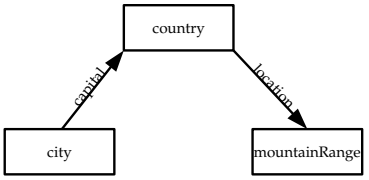
1

INTRODUCTION

1.1 MOTIVATION

As the amount of formalized conceptual knowledge available through datasets and ontologies grows, there is an increasing need to make this knowledge accessible to humans in an easy and intuitive way. One way to accomplish this is by means of language technology, e.g. in the form of question answering systems, that allows users to query repositories of conceptual knowledge through natural language. One of the main challenges in building such language technology systems is mapping the natural language input — whether written or spoken — onto some kind of formal representation that can be used to query the underlying knowledge base. As an example, consider the small ontology on the domain of geography, plus the corresponding dataset, given in figure 1. Boxes represent ontology classes, while edges represent properties, i.e. relationships between classes. A question answering system that builds upon this ontology and triple store may receive the following question as input, and turn it into the subsequent SPARQL query in order to retrieve the correct answer from the triple store:

Example 1.1.1. *What is the capital of France?*




```
France isA country.
Paris isA city.
MassifCentral isA mountainRange.
Paris capital France.
France location MassifCentral.
```

Figure 1: Example ontology and data set

```
SELECT ?c WHERE {
        ?c capital France .
}
```

In this case, the mapping between the natural language input and the SPARQL query seems straightforward: The system may detect *capital* and *France* as important keywords and search the underlying knowledge base for elements that are labeled by these same terms. Here, it would turn out that `capital` is an ontology property that links the classes `city` and `country` and that `France` is an individual of the class `country`, and based on this information the above SPARQL query may be generated. Most of the time, however, mapping between the natural language input and formalized representations will not be that easy, as there will often be some kind of mismatch between the terms used in the natural language input and those used in the knowledge base. As an example, consider the following input and query pair:

Example 1.1.2. *In which countries does the Massif Central lie?*

```
SELECT ?c WHERE {
        MassifCentral location ?c .
}
```

Here, the system needs some way of knowing that the natural language terms *countries* and *lie* map to the class `country` and the property `location`, respectively, which is not evident from the terms itself. This problem is even more severe when the input language and the language the knowledge base identifiers are given in do not match: For example, a German input question such as *Wie heißt die Hauptstadt von Frankreich?* in itself does not contain any clue to the system that it should be mapped to the SPARQL query given in example 1.1.1.
Therefore, language technology systems that build upon repositories of conceptual knowledge need access to information on how the elements of the repository at hand can be verbalized in a given language. Ontology languages support the inclusion of such information to a certain extend, e.g. by means of `rdfs:label` or SKOS properties. However, these ontology-internal mechanisms usually do not provide further information about the labels' linguistic behavior, such as their part-of-speech or irregular inflectional forms they may take. In addition, labels only capture one canonical way of verbalizing an ontology element, but do not provide lexical variants. For example, in case of the ontology given in figure 1 the property `location` may be verbalized in

English both as *to be located in* and *to lie in*, and we would want to capture information about both of these linguistic variants. Furthermore, in actual knowledge bases the coverage provided by such labels is often very limited, in particular for languages other than English: In case of the Wikipedia-based dataset DBpedia (Auer et al. 2007), for instance, the majority of individuals do not have any language label at all, and while most other ontology elements are assigned an English label, coverage for other languages is very restricted; for example, only around ten percent of DBpedia's classes and properties have a Japanese label, as shown in table 1.

As a result, in many scenarios external resources of linguistic information will be preferable in order to make resources of conceptual information accessible to language technology systems. However, lexical resources such as Wiktionary[1] or WordNet(Miller 1995), while providing rich linguistic information and lexical variants, do not contain any anchors between verbalizations and elements of a specific ontology.

One further type of lexical resource are ontology lexicons (Prévot et al. 2010, McCrae et al. 2011b), which were specifically designed for the task of linking ontology elements to possible verbalizations in a given language enriched with various kinds of linguistic information. Conventionally, such ontology lexicons are generated manually, which is a very time-consuming task that requires expertise in Semantic Web technologies and lexicon engineering, as well as knowledge about the domain of the ontology at hand. Furthermore, in order to decide which verbalizations are appropriate for a given ontology element, in many cases one either needs to have a very good command of the target language at hand oneself, or one should at least be able to consult with native speakers, which in case of smaller target languages may pose a problem. While the latter problem may in principle be solved by translating an already existing ontology lexicon automatically (McCrae et al. 2011a, Arcan and Buitelaar 2013), corresponding systems have not yet reached an accuracy sufficient to produce high-quality lexicons off the shelf. Furthermore, with this approach one may often not retrieve the best possible verbalization of a given ontology element in the target language, for example when the target language provides verbalizations whose semantic granularity fits closer to the semantics of the ontology element at hand than any verbalization available in the source language.

Therefore, this thesis will deal with two alternative approaches to ontology lexicalization that require less manual effort: On the one hand, we will look at M-ATOLL (Walter 2017), a framework for the (semi-)automatic generation of ontology lexicons in the RDF-based lemon format (McCrae et al. 2011b); on the other hand, we will investigate

[1] `https://www.wiktionary.org`

Language	Individuals	Classes, properties	Total
English	37.72%	99.97%	37.78%
French	9.16%	21.13%	9.17%
German	7.87%	57.63%	7.91%
Italian	7.42%	6.84%	7.42%
Dutch	7.12%	35.62%	7.14%
Spanish	7.00%	7.23%	7.00%
Polish	6.57%	3.07%	6.57%
Portuguese	5.84%	6.22%	5.84%
Russian	5.63%	0.53%	5.62%
Chinese	4.18%	0.31%	4.18%
Japanese	3.69%	10.44%	3.69%
Arabic	2.05%	0.11%	2.05%

Table 1: Percentages of ontology elements for which respective language label is available within the DBpedia ontology. For the sake of brevity, only languages for which labels on individuals are available are given.

whether crowdsourcing (Howe 2006) can be used to generate ontology lexicons of good quality at acceptable costs. In recent years crowdsourcing has already been used for a number of different tasks related both to natural language processing and ontologies Snow et al. (2008), Ambati and Vogel (2010), Acosta et al. (2013); however, to our knowledge prior to Lanser et al. (2016) there existed no reports on using crowdsourcing specifically for ontology lexicalization.

Another main topic of this thesis is ontology lexicalization specifically for Non-Indo-European languages: So far, M-ATOLL has been used with a number of Indo-European languages — English, German and Spanish — that share a rather large set of common characteristics. We will investigate whether M-ATOLL can also be used fruitfully with Non-Indo-European languages and what kinds of adaptations to the framework would be necessary in order to make that work. As mentioned before, generating ontology lexicons for smaller target languages can be a problem if not enough proficient speakers are available to provide input on which verbalizations would be appropriate. We will look at whether crowdsourcing can help to overcome such a lack of speaker input. Finally, we will test whether lemon, the format for the specification of ontology lexicons used by M-ATOLL, in itself is flexible enough to support ontology lexicons in Non-Indo-European languages.

In order to investigate these topics we will use both M-ATOLL and crowdsourcing to

generate Japanese ontology lexicons in the lemon format for excerpts from DBpedia's ontology. There already exists an English ontology lexicon for DBpedia (Unger et al. 2013), which would allow us to compare both the end results and the proceeding. We chose Japanese as our example language as it is one of the few Non-Indo-European languages for which a comparably large amount of NLP-related tools and resources as required by M-ATOLL is available. While working with a more underresourced language and seeing how the problems emerging from data sparseness in this case may be solved would definitely be worthwhile, in the context of this thesis we wanted to focus on problems with language portation that are more directly related to the structure of M-ATOLL and lemon, respectively.

1.2 TASK DEFINITION

As discussed in the previous section, ontology lexicons have a number of advantages over other means of providing linguistic information on ontology elements, such as ontology-internal mechanisms like `rdfs:label`. Since generating ontology lexicons by hand is a very labor-intensive task, in this thesis we want to find alternative approaches to creating ontology lexicons for Non-Indo-European languages. Such alternative approaches should meet the following criteria:

- The manual effort required by each individual worker involved in the ontology lexicalization process should be as minimal as possible. This concerns both the workers involved in the crowdsourcing approach as well as the pre- and postprocessing stages of both approaches.
- The overall costs — i.e. the effort, time and money that need to be spent on ontology lexicalization with the approach at hand — need to be acceptable.
- The entries generated by means of the approach at hand should be of appropriate quality, i.e. they need to contain meaningful verbalizations, plus correct and sufficient linguistic information.
- The approach should be suitable for smaller languages, for which finding enough speakers may be difficult.

We look at two such alternative approaches in particular, and test whether they meet these criteria:

- Crowdsourcing can be defined as "the act of taking a job traditionally performed by a designated agent (usually an employee) and outsourcing it to an undefined, generally large group of people in the form of an open call" Howe (2006). Outsourcing work this way is done via a crowdsourcing platform, such as Amazon MechanicalTurk[2] or CrowdFlower[3], where employers upload jobs and determine how much they want to pay the workers, and workers browse job offerings and decide which of the jobs they want to work on. Crowdsourcing is often used for tasks that can be split up into lots of small so-called microtasks, so that each worker only works on a subset of the overall task. As crowdsourcing gives access to a very large potential workforce, work can often be done much faster and at

[2] `https://www.mturk.com/`
[3] `http://www.crowdflower.com/`

lower costs, and it can be easier to perform large-scale projects where lots of data needs to be collected. Furthermore, this access to a large workforce also makes it easier to find workers with certain specialized skills, such as certain language skills, which at least in case of smaller languages may be difficult otherwise.

- M-ATOLL (Multilingual, Automatic inducTion of OnToLogy Lexica) is a framework for the (semi-)automatic generation of ontology lexicons that combines three different approaches. The main approach, which will be dealt with in this thesis, is the corpus-based one, which works for ontology properties only. It requires a triple store based on the ontology at hand, which gets searched for all triples `s p o` where `p` is the property under consideration. Then for each triple a corpus gets searched for sentences that contain references to the subject and object pair `s` and `o`. The sentences retrieved this way are dependency-parsed, and handwritten dependency patterns are applied to the sentences to extract elements that often express a relation between the triple subject and object. For example, for a copula construction such as *Paris is the capital of France* there would be a dependency pattern that extracts *capital*. While requiring additional external resources such as a dependency parser and a corpus, the main advantage of this approach is that it is independent of the labels provided by the ontology itself, and can hence also be used for ontology elements for which no label is provided by the ontology at all. The resulting ontology lexicon is serialized by means of the lemon model.

The lexicons generated by either of these approaches should fulfill the following requirements:

- The specification format should support multilinguality; in particular, it should be flexible enough to be able to represent linguistic variance e.g. with respect to part-of-speech systems, syntactic constructions etc. among as many different languages as possible.
- In principle it should be possible to specify finer semantic differences among verbalizations that go beyond the ontology elements a given verbalization may refer to. This may be particularly important in cases where the language the ontology identifiers are specified in and the lexicon's target language do not match.

As an instantiation of our approach we will use both crowdsourcing and M-ATOLL to generate Japanese lemon lexicons for excerpts from DBpedia's ontology.

1.3 RESEARCH QUESTIONS

1. Are crowdsourcing and M-ATOLL appropriate for creating ontology lexicons in Non-Indo-European languages of good quality and at acceptable costs?
2. How do the two approaches compare to each other with respect to their costs and the quality of the resulting lexicons?
3. How do the results of M-ATOLL for Japanese compare to the results for English?
4. Are the external tools and resources required by M-ATOLL available for Non-Indo-European languages? If they are not, or if they differ considerably from corresponding tools and resources for Indo-European languages, how does one cope with that?
5. Is lemon, the format for the specification of ontology lexicons used by M-ATOLL, flexible enough to represent information on the linguistic properties of Non-Indo-European languages as required by ontology lexicons, even if these differ considerably from those of Indo-European languages?
6. Is lemon able to represent finer semantic differences among verbalizations that go beyond the ontology element they refer to, such as in cases where there is a degree of mismatch between the exact semantics of the verbalization and the ontology element at hand?
7. Are our handwritten dependency patterns, which are employed by M-ATOLL's corpus-based approach to extract verbalizations, of sufficient quality?
8. Are there alternative, (semi-)automatic approaches to the generation of these patterns with promising results?
9. Which parts of M-ATOLL are language-specific and accordingly need to be adapted to the characteristics of new target languages?
10. Is crowdsouring a feasible way of overcoming a lack of available speakers in case of smaller languages?

1.4 PUBLICATIONS

Parts of this thesis have been published before in the following paper:

Bettina Lanser, Christina Unger, and Philipp Cimiano. Crowdsourcing ontology lexicons. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asunción Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*. European Language Resources Association (ELRA), 2016. URL `http://www.lrec-conf.org/proceedings/lrec2016/summaries/217.html`

2

PRELIMINARIES

2.1 OVERVIEW

This chapter introduces foundations crucial to the understanding of this thesis and provides some further information on some of the topics touched upon in this work. Section 2.2 deals with lexicography and the similarities and differences between resources of linguistic and conceptual knowledge, which is relevant to understanding the relationship between ontologies and resources that are supposed to lexicalize them, such as ontology lexicons. Next, section 2.3 gives a short introduction to the topic of crowdsourcing. The following section 2.4 provides some information on lemon, a data model for machine-readable lexicons, which will be used for representing the Japanese ontology lexicons for DBpedia whose generation will be the main topic of this thesis, while section 2.5 deals with M-ATOLL, the system for the semi-automatic induction of ontology lexicons that together with crowdsourcing forms one of the approaches to the generation of ontology lexicons under consideration in this thesis. Section 2.6 provides a short overview over the Japanese language and some of its grammatical characteristics that may be relevant to the induction of Japanese ontology lexicons. Finally, since one of the approaches included in M-ATOLL makes use of dependency relations in order to extract verbalizations of ontology elements from texts, in section 2.7 we give an introduction to dependency grammar and parsing.

2.2 COMPUTATIONAL LEXICOGRAPHY

2.2.1 *Overview*

According to Gibbon (2000, p. 2), "[l]exicography is the branch of applied linguistics concerned with the design and construction of lexica for practical use". A lexicon can be defined as an index that maps from a list of words in a language — a vocabulary — to information about the characteristics and use of the respective word (Hirst 2009, p. 1). Computational lexicography, then, deals with lexicons for computational use and the computational methods which are used to create or process these resources. As such, the field is closely interlinked with other subdisciplines of computational linguistics, in particular any subdiscipline that relies on structured representations of linguistic information, such as machine translation. Computational lexicons can be meant either for human use or NLP applications; in general, lexicons intended for humans are unsuitable for being used in NLP without substantial revision due to their style and format, and vice versa (Spohr 2012).
As already mentioned in chapter 1, compiling lexicons by hand is a very labor-intensive and time-consuming task. Hence, a number of alternative approaches to the generation of lexicons for NLP applications have emerged: On the one hand, while in themselves unsuitable for NLP tasks, already existing electronic lexicons for human use can serve as the basis for the development of an NLP lexicon. One example, of a resource developed this way would be ACQUILEX I[1] (Briscoe et al. 1993). On the other hand, it is also possible to infer lexical information from the usage of words as observed in text corpora (e.g. Light (1998)). Furthermore, because of the many regularities in the ways that natural languages generate derived words, many of the entries in a lexicon can be automatically predicted. For example, Viegas et al. (1996) present a system that proposes candidate entries for an English lexicon by adding inflection and affixation to words, such that e.g. *read* yields additional entries for *reader*, *unreadable*, *readership*, and so on. A lexicographer must then filter the proposals.

2.2.2 *Lexicons and Ontologies*

Ontologies can profit from being enriched with linguistic information such as that found in lexicons in a number of ways (Cimiano et al. 2011, p. 2): Apart from the

[1] `https://www.cl.cam.ac.uk/research/nl/acquilex/acqhome.html`

potential benefits when using a given ontology within an NLP application, during ontology engineering developers will be able to better understand and manipulate an ontology if information is available about how its elements are verbalized in a given language. Furthermore, lexicons and ontologies share many similarities: On the one hand, the meanings of the lemmas contained in a lexicon can be understood to correspond to a category or concept within an ontology, a position that is particularly appealing in case of lexicons for NLP applications where the deeper philosophical problems of the notion of meaning can be ignored. In addition, the relationships holding between the meanings in a lexicon and the relations between concepts in an ontology are very similar to one another; for example, the lexical relation of hyponymy is very close to the `is-a` relation found in ontologies. Therefore, one may wonder whether two separate kinds of resources, such as ontologies and ontology lexicons, are really necessary, or whether it would not be better to merge both kinds of resources into one. Furthermore, one may even ask if lexicons may not be used as ontologies. However, while lexicons and ontologies share many structural similarities, generally a lexicon does not make a good ontology, and vice versa, and one should therefore not try to merge both kinds of resources into one: An ontology is a set of categories of objects in the world and relationships among them, and therefore a non-linguistic object, while a lexicon, by definition, depends upon a natural language and the word senses found in it. This difference leads to a number of points in which lexicons and ontologies substantially differ from each other (Hirst 2009, p. 8-13):

- In case of ontologies, it is usually assumed that the subcategories of a common category are disjoint. Hence, in case of a category `pet` with subcategories `dog` and `cat`, it is assumed that no `pet` is a `dog` and a `cat` at the same time. However, in natural languages the meaning of terms with a common hypernym regularly overlap in this way; one says in these cases that the terms are near-synonyms. For example, the English words *error* and *mistake*, while differing in their connotations, overlap in their meaning this way (Hirst 2009, p. 8). Furthermore, the differences between members of such near-synonym clusters differ between languages. For example, in case of the *error/mistake* cluster German has the word *Irrtum* that refers to mistakes confined to the mind, not embodied in something done or made, a distinction that is not made in English (Edmonds and Hirst 2002, p. 112). Hence, the senses of such words do not map well onto an ontology.

- A lexicon, by definition, does not contain references to concepts that are not lexicalized in the given language. The words of any language, when taken to

represent ontological categories, are merely a subset of the categories that would be present in a complete ontology of a given domain. On the one hand, every language has lexical gaps with respect to other languages, i.e. concepts that it does not lexicalize but the other language does. On the other hand, there may also be ontological categories that are not lexicalized in any language at all, given that natural languages tend to preferably lexicalize those concepts that are most salient and important to human daily life, which are commonly found in the middle of the ontological hierarchy. Hence, natural languages omit many distinctions that one would like to make in an ontology.

- However, there are also semantic distinctions made in natural language one would probably not want in an ontology, or which are not even a reliable reflection of the world. For example, in Japanese nouns cannot be directly modified by numerals, but must be quantified by so-called counters consisting of the numeral and a classifier, as shown in the following examples:

  (1) *二-犬
  ni-inu
  two-dog
  *two dogs

  (2) 二-匹-の　犬
  ni-biki-no　*inu*
  two-CL-POSS　dog
  two dogs

  Japanese has various classifiers, and each one of them imposes semantic restrictions on its objects. For example, the classifier 匹 (*biki*) used in the example above can only be used for small animals. Overall, Japanese has the following classifiers for animals (Ebi and Eschbach-Szabo 2015, p. 120):

  – 匹 (*biki*) for small animals
  – 頭 (*tou*) for large animals
  – 羽 (*wa*) for birds

  While the distinction between birds and other kinds of animals, plus the distinction between non-bird animals based on their size, is clearly semantic, it is unlikely that these are the kinds of distinctions one would want to make in a practical ontology (Hirst 2009, p. 11).

For these reasons, merging of ontologies and lexicons into one kind of resource, or even the substitution of ontologies with lexicons, would not be desirable. However, due to the structural similarities described above, ontologies and lexicons are resources that can still fruitfully build on each other. Using ontologies together with ontology lexicons provides an ideal way of enriching ontologies with lexical information while keeping lexical and conceptual information separate, and thus avoiding many of the problems that may emerge from the discrepancies between linguistic and conceptual information just described.

Apart from ontology lexicons, there have been further approaches to building one kind of resource from the other: On the one hand, there have been attempts to derive ontologies, or at least semantic hierarchies, from lexicons (e.g. Amsler (1981), Richardson et al. (1998), Philpot et al. (2005)). On the other hand, an ontology may provide an interpretation or grounding of word senses as found in a lexicon, e.g. by representing word senses as references to elements in the ontology as it is also done in case of ontology lexicons. An example of this approach would be the Unified Medical Language System[2] (UMLS) (Lindberg et al. 1993), which contains a so-called Metathesaurus whose concepts together constitute an ontology, and each such concept is annotated with a set of terms in English and other languages that can be used to denote it. A field where such a mapping between word senses and an ontology may be of particular interest is machine translation and other multilingual applications: Here, the ontology could serve as a kind of interlingua, allowing one to first transfer words from the source language into ontological concepts and then to transfer these concepts into words from the target language. An example lexicon suitable for this would be SIMPLE (Bel et al. 2000), a twelve-languages lexicon whose word meanings are presented by means of a hand-crafted upper ontology (Hirst 2009).

[2] https://www.nlm.nih.gov/research/umls/

2.3 CROWDSOURCING

Crowdsourcing is a relatively recent concept that encompasses many different practices. Accordingly, there exist varying definitions, and practices that are classified by some authors as a form of crowdsourcing are not classified as such by others. One of the definitions most cited in the literature is the one by Jeff Howe, who coined the term and defined it as "the act of taking a job traditionally performed by a designated agent (usually an employee) and outsourcing it to an undefined, generally large group of people in the form of an open call" (Howe 2006). The development of crowdsourcing was sparked on the one hand by today's 24/7 access to massive human crowds through the internet and on the other hand by the fact that there are still a large number of data processing operations that are difficult or impossible to fully automate, such as tasks involving language or image understanding or polling people for their subjective opinion on some topic. There are a number of distinct forms of crowdsourcing which can be classified according to a number of different parameters. For example, based on the motivation of the crowdsourcing workers one can differentiate between volunteer-driven crowdsourcing, which provides non-financial incentives to the workers such as enjoyment or prestige and includes e.g. citizen science sites such as GalaxyZoo[3] or eBird[4], and commercial crowdsourcing services, where the main motivation for workers to participate is financial compensation. Commercial crowdsourcing is usually carried out via specific crowdsourcing platforms, the most prominent one probably being Amazon Mechanical Turk, on which individuals or organizations who have work to be performed, the so-called requesters, post tasks and define how much they want to pay workers, and people who signed up to the platform as workers may choose to complete the tasks. Each unit of work to be performed by a worker is called a human intelligence task (HIT) or microtask. For example, given that a researcher would like to label 100 images and creates a corresponding experiment on a crowdsourcing platform, the researcher would be the requester, each person choosing to label an image is a worker and an individual HIT would consist of labeling a single image. Crowdsourcing platforms were originally developed to allow companies to outsource work, but are now also used productively for research. Due to their reliance upon large amounts of labeled language data, two of the first research fields to adopt crowdsourcing were NLP and machine translation, and language processing was one of the first large-scale uses of crowdsourcing technologies (Munro and Tily 2011).

[3] https://www.galaxyzoo.org/
[4] http://ebird.org/content/ebird/

2.4 LEMON

lemon (Lexicon Model for Ontologies) (McCrae et al. 2011b) is a data model for machine-readable lexicons that allows lexical information to be represented relative to an ontology and to be shared on the Semantic Web. Since lemon is based on RDF and the principles of Linked Data, lemon lexicons can be easily extended, reused and linked to each other. It has been published as a W3C vocabulary and is currently under consideration to be standardized under the W3C[5]. lemon does not rely on a certain inventory of linguistic categories or a specific grammar framework. Instead, it supports the reuse of any already existing linguistic ontology, such as OLiA (Chiarcos 2012) or ISOcat (Kemps-Snijders et al. 2008), and parts-of-speech, syntactic frames and argument roles have to be imported from such an linguistic ontology. M-ATOLL chooses this ontology to be LexInfo (Cimiano et al. 2011).

lemon is modular in nature; the architecture of the core module is shown in figure 2. A Lexicon contains a set of Lexical Entries and is marked with a language tag, and hence mono-lingual. The central element in lemon is the Lexical Entry, which all other elements are connected to and which represents a single term. Each Lexical Entry consists of a number of Lexical Forms, which correspond to inflectional variants of the respective term and may be marked as `canonical` — this form corresponds to the entry's lemma — `other` or `abstract`. lemon assigns semantics to lexical entries by means of references to ontology elements. Therefore, every entry is mapped to entities in a given ontology by means of Lexical Sense objects. This is a one-to-many relation: A given ontology element can be the reference of several Lexical Entries, and a given Lexical Entry can have several Lexical Senses and hence ontology references.

Lemon-based ontology lexicons are serialized as RDF (Resource Description Framework), which is a W3C standard first presented in 2004 that supports the exchange of data on the web and describes information that is implemented in web resources in a structured and queryable way. The RDF data model is based on the idea of making statements about so-called resources, which are identified by Uniform Resource Identifiers (URIs). For example, the DBpedia property `spouse` would be represented by the URI `http://dbpedia.org/ontology/spouse`. Statements about resources are given as so-called triples of the form

```
subject predicate object .
```

[5] `http://www.w3.org/2016/05/ontolex/`

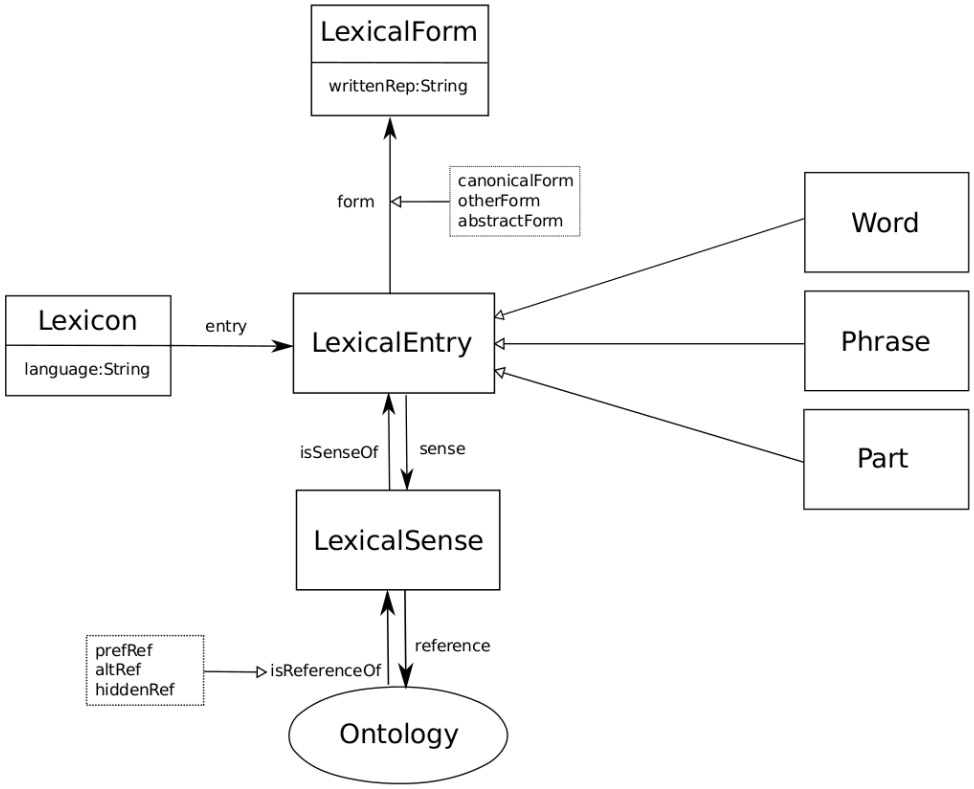


Figure 2: Architecture of lemon's core module

where subject and predicate are resources and object is either also a resource or a literal, such as a string. For example, with respect to elements found in DBpedia's ontology, the triple

```
<http://dbpedia.org/resource/Don_Quixote>
        <http://dbpedia.org/ontology/author>
                <http://dbpedia.org/resource/Miguel_de_Cervantes> .
```

expresses that the relationship `author` holds between the entities `Don_Quixote` and `Miguel_de_Cervantes`, while a triple such as

```
<http://dbpedia.org/resource/Cobble_Hill_Tunnel>
        <http://dbpedia.org/ontology/yearOfConstruction>
                "1844"^^<http://www.w3.org/2001/XMLSchema#gYear> .
```

expresses that the entity `Cobble_Hill_Tunnel` has the value `1844` with respect to the property `yearOfConstruction`. A set of triples represents a graph, where the edges correspond to the predicates and the nodes correspond to the subjects and objects. This graph can be serialized in a number of formats; as an example, consider the following lemon lexicon entry for *wife*, given in Turtle notation[6]. Morphosyntactic information is specified by means of the LexInfo vocabulary:

```
:wife a lemon:Word ;
        lexinfo:partOfSpeech lexinfo:noun ;
        lemon:canonicalForm :wife_canonicalForm ;
        lemon:synBehavior :wife_synBehavior ;
        lemon:sense :wife_sense .
:wife_canonicalForm lemon:writtenRep "wife"@en .
:wife_synBehavior a lexinfo:NounPPFrame ;
        lexinfo:subject :ssyn ;
        lexinfo:directObject :osyn .
:wife_sense lemon:reference <http://dbpedia.org/ontology/spouse> ;
        lemon:subjOfProp :ssyn ;
        lemon:objOfProp :osyn .
```

Multilinguality can be modeled in lemon through lexical entries in lexicons of different languages having the same reference in a given ontology. This way, one can publish translated versions of a given lexicon without needing to modify the original lexicon or the original ontology (McCrae et al. 2011b). As an example, consider the following

[6] `http://www.w3.org/TeamSubmission/turtle/`

Japanese entry for 妻 (*tsuma*; 'wife'), which shares with the English entry given above the reference to the URI <http://dbpedia.org/ontology/spouse>:

```
:妻 a lemon:Word ;
        lexinfo:partOfSpeech lexinfo:noun ;
        lemon:canonicalForm :妻_canonicalForm ;
        lemon:synBehavior :妻_synBehavior ;
        lemon:sense :妻_sense .
:妻_canonicalForm lemon:writtenRep "妻"@ja .
:妻_synBehavior a lexinfo:NounPPFrame ;
        lexinfo:subject :ssyn ;
        lexinfo:directObject :osyn .
:妻_sense lemon:reference <http://dbpedia.org/ontology/spouse> ;
        lemon:subjOfProp :ssyn ;
        lemon:objOfProp :osyn .
```

2.5 M-ATOLL

M-ATOLL (Multilingual, Automatic inducTion of OntoLogy Lexica) (Walter 2017) is a framework for the automatic induction of ontology lexicons in multiple languages. In general, the framework takes as its input at least an ontology, together with a corresponding knowledge base, and produces as its output a lexicon serialized in the lemon format that lexicalizes the input ontology. M-ATOLL is a combination of three different approaches:

- The label-based approach, which works both for ontology classes and properties,
- the corpus-based approach, which lexicalizes properties only,
- and an approach based on machine learning that deals with adjective verbalizations of properties.

Figure 3 shows a visualization of the first two approaches. The label-based approach, shown on the left, makes use on the one hand of the ontology-internal labels of classes and properties, and external lexical resources on the other hand: First, for the given class or property and a given language the corresponding label is extracted from the knowledge base, i.e. for a given ontology element `e` and a given language `l` one looks for a triple of the form

```
e rdfs:label o .
```

with `lang(o)=l` and extracts this triple's object `o`. Then, one retrieves all synonyms of that label from some external lexical resource; for example, one may search WordNet for synsets that contain the given label. In case of classes, these synonyms are further filtered to only keep the most relevant ones based on a synonym selection algorithm inspired by Lesk (1986). Finally, a number of lexical entries are created from the label itself and its (most relevant) synonyms by means of predefined templates.
The corpus-based approach, which is M-ATOLL's main approach, is based on a dependency-parsed text corpus whose sentences M-ATOLL tries to match to predefined, language-specific linguistic patterns. It consists of two main steps: First, M-ATOLL tries to extract relevant sentences from the corpus that may express a given property `p`, and preprocesses the sentences retrieved this way, as follows: First of all, for the given property `p` all triples are extracted from the knowledge base that contain this property as their predicate, i.e. which have the form

```
s p o .
```

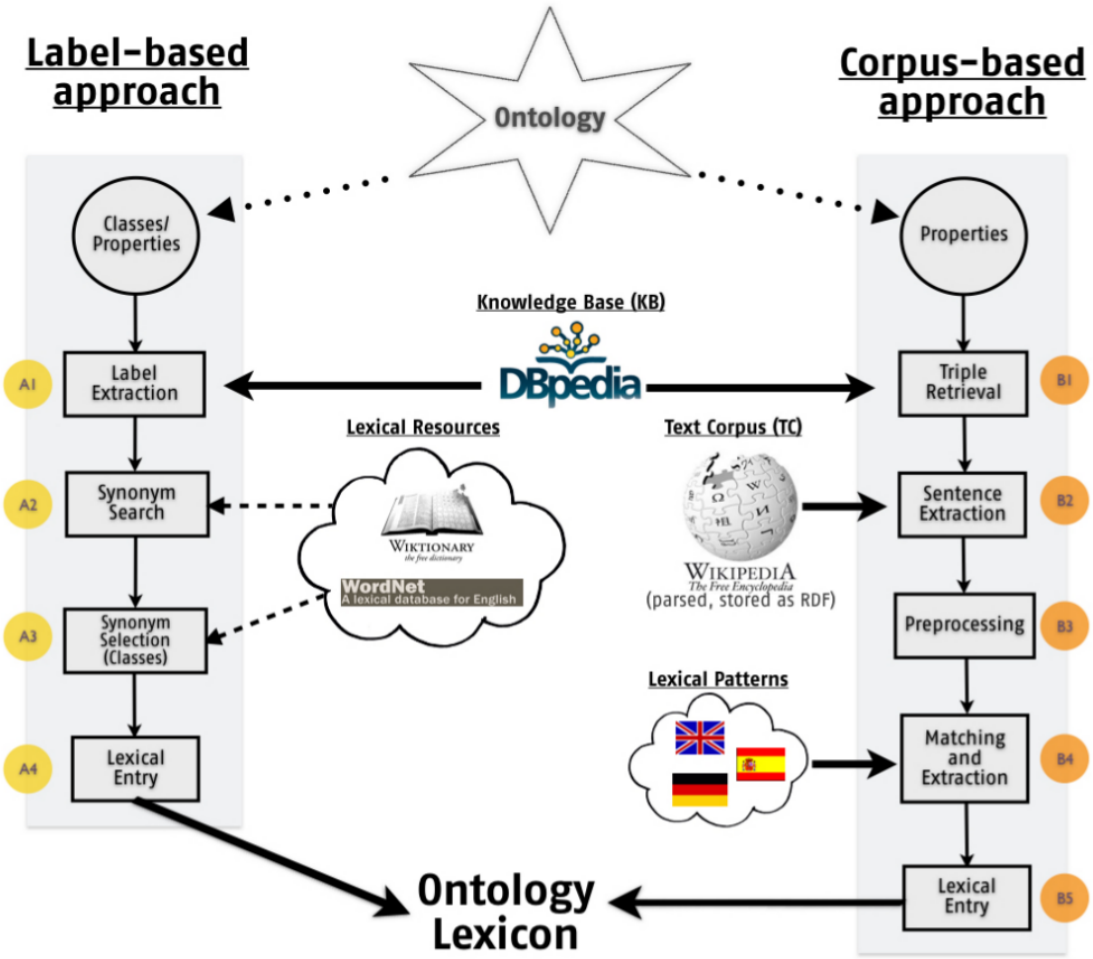


Figure 3: Label- and corpus-based approach of M-ATOLL (Walter 2017, p. 44).

Then, for each subject-object pair `s,o` retrieved this way, one selects all those sentences from the corpus for further processing that contain labels of the subject and object, i.e. which contain strings `s',o'` such that

```
s rdfs:label s' .
```

and

```
o rdfs:label o' .
```

are contained in the knowledge base. The dependency parses of the sentences which have been selected this way are then converted into RDF, and the nodes in the dependency tree that correspond to the subject and object labels based on which the sentence was selected are marked. In the second step of the corpus-based approach, the actual candidate lexicalizations are extracted from the sentences which were selected and turned into RDF in the preceding step: Each selected sentence is matched against a set of handcrafted, language-specific dependency patterns specified as queries in the SPARQL Query Language for RDF[7]. Since the sentences are given in RDF, this amounts to a simple query operation. If there is a match between a sentence and a dependency pattern, a lexical entry is created. To do so, the output of the SPARQL query is matched onto one of several lemon-based templates. Finally, the candidate lexical entries retrieved this way are filtered, e.g. based on the number of sentences they were encountered in, in order to reduce noise in the final lexicon, and the actual lexicon is serialized as lemon RDF. Since the corpus-based approach works independently from the labels of a given property, one can generate lexicon entries also for properties with missing or nondescript names such as `property01`, which are otherwise difficult to verbalize even for humans.

As an example of how M-ATOLL's corpus-based approach works, consider we want to find English verbalizations for DBpedia's `spouse` property, as shown in figure 4: First, M-ATOLL would search DBpedia's triple store for triples with `spouse` as their predicate. Among others, it would find the triple `Barack_Obama spouse Michelle_Obama`. Hence, in the next step, for this one triple it would search the dependency-parsed text corpus for sentences that contain labels of the ontology individuals `Barack_Obama` and `Michelle_Obama`, such as the English labels *Barack Obama* and *Michelle Obama*. As a result, it might find the dependency parse of the sentence *Michelle Obama is the wife of Barack Obama*, which would be selected for further processing and be converted into RDF. In the next step, the RDF version of this sentence would be matched against

[7] `https://www.w3.org/TR/rdf-sparql-query/`

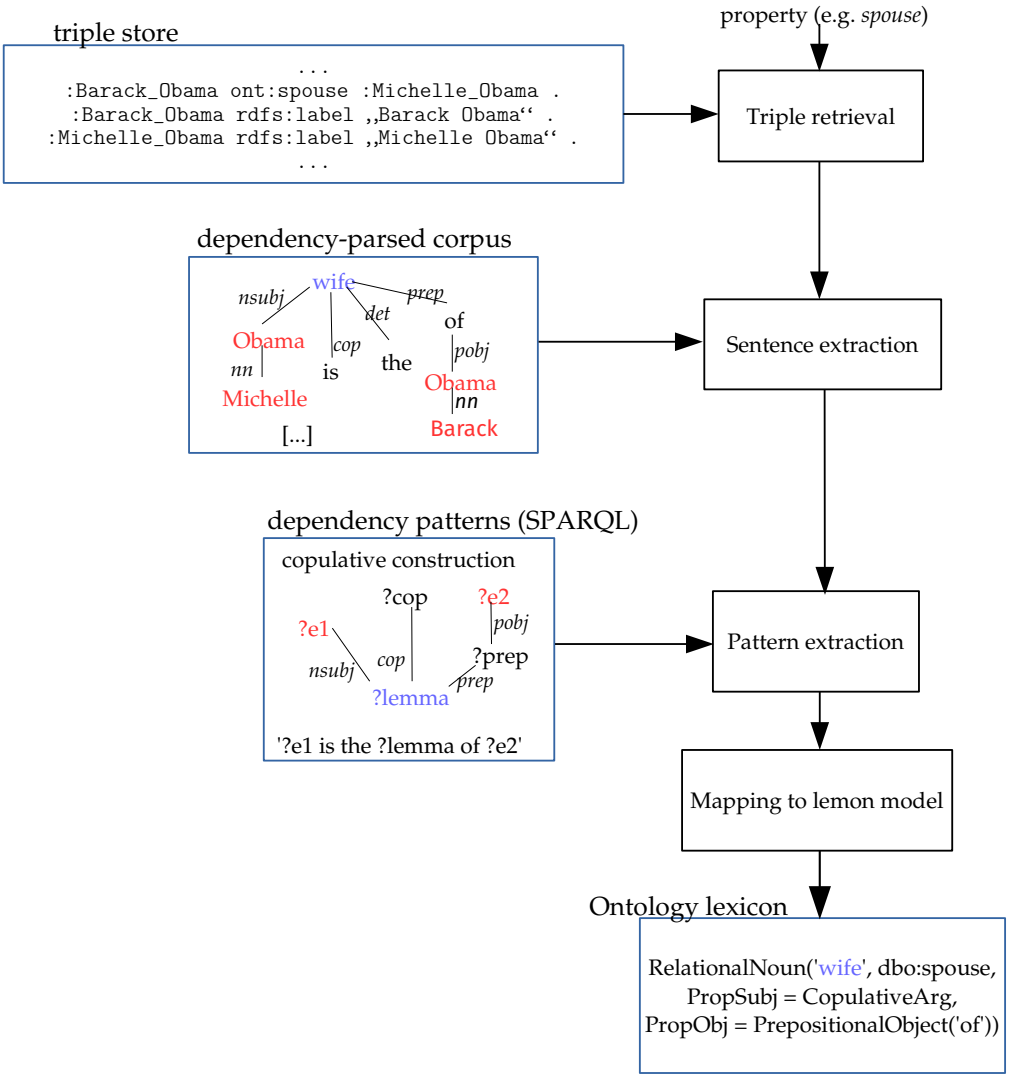


Figure 4: Example run of M-ATOLL's corpus-based approach

the dependency patterns available for English, and among others, it may be matched against a dependency pattern that corresponds to

```
[subj] is the [lemma] of [obj]
```

This pattern would be a match, and the element in the `[lemma]` position of the pattern — *wife* — would be turned into a candidate lexicon entry for the property `spouse`.
The third, machine-learning-based approach again only works for properties. For a given property all triples are extracted from the knowledge base that contain the property as their predicate and an adjective in their object position. Then a machine learning approach is applied to the set of selected objects to find those adjectives that may instantiate a correct verbalization of the given property.
So far, all three approaches covered by M-ATOLL support English, while the corpus-based approach also supports German and Spanish. Similarly, since it is the core approach of M-ATOLL, in this thesis we will first adapt M-ATOLL's corpus-based approach to Japanese.
A lexicalization generated by M-ATOLL for an ontology element `E` in a language `L` comprises the following core information:

- a lemma in `L` (the actual verbalization)
- syntactic information:
  - the lemma's part-of-speech category (e.g. verb)
  - a subcategorization frame that specifies the required syntactic arguments (e.g. transitive)
- semantic information:
  - a reference within the ontology (i.e. `E`)
  - a mapping of the semantic arguments of the ontology element to the syntactic arguments of the lemma

For example, in case of the *wife* example given above, the resulting lexicon entry would have the following form:

- lemma: *wife*
- syntactic information:
  - part-of-speech: noun

  - frame: NounPP
    arguments:
    * copulative subject s_{syn}
    * prepositional object o_{syn} with marker *of*

- semantic information:
  - reference: `spouse`
  - mapping:
    * subject $\mapsto s_{syn}$
    * object $\mapsto o_{syn}$

2.6 JAPANESE

Japanese has around 122 million first-language speakers, which makes it the world's ninth largest language in terms of the number of speakers[8]. Based on a combination of the number of speakers, distribution, internationality and prestige of languages, Weber (1997) lists Japanese as the eighth most influential language of the world; a newer study, which ranks the languages of the world based on their share in global information production — i.e. according to the number of book, newspaper and magazine publications, film productions and webpages — sees Japanese on rank 6 behind English, German, Spanish, Chinese and French (Lobachev 2008). Together with the Ryukyuan languages it forms the Japonic language family, whose relation to other language groups is unclear; some argue for a relationship to the Altaic languages, others see the Japonic language family as an isolate. Japanese is for the most part an agglutinative language, i.e. words may consist of several morphemes, the morphemes usually remain unchanged when merged together to form a word, and each morpheme carries at most one grammatical category as its meaning. As an example, consider the following verb form, consisting of the stem *eat* and suffixes expressing the causative, passive, honorative and the past tense, respectively:

(3) 食べ-させ-られ-まし-た
tabe-sase-rare-mashi-ta
eat-CAUSE-PASS-HON-PAST
[I] was made to eat [something].

(Ebi and Eschbach-Szabo 2015, p. 19)

The Japanese writing system comprises three distinct scripts which in texts are generally used together: The — logographic — Chinese characters, called kanji (漢字) in Japanese, and the syllabic scripts hiragana (ひらがな) and katakana (カタカナ), which developed from kanji. Kanji is generally used for the stems of nouns, verbs and adjectives, hiragana for the grammatical affixes added to the stem, and katakana, among other usages, is used for foreign and loan words, as can be seen in the following example:

(4)

私-は	アイスクリーム-を	食べました。
watashi-wa	*aisukuriimu-o*	*tabemashita*
I-TOP	ice.cream-DOBJ	ate

I ate ice cream.

[8] http://archive.ethnologue.com/16/ethno_docs/distribution.asp?by=size

Furthermore, Japanese texts can also contain Latin letters (rōmaji) e.g. in form of international acronyms such as *EU* (Ebi and Eschbach-Szabo 2015, p. 50).
The Japanese inventory of sounds is comparably small with only 23 consonants and 5 vocals. There are only monophthongs, no diphthongs. Vowel length is phonemic, for each vowel there is both a long and a short version.
One interesting aspect of the Japanese part-of-speech inventory is that there is not one single adjective part-of-speech. Rather, there are at least two different types of words that behave semantically like adjectives:

- i-adjectives end on -i and for the most part belong to the native Japanese stratum. They can constitute the sentence's predicate on their own, which makes them similar to verbs. For this reason, they are also called verb-like adjectives.

  (5) 像-の 鼻-が 長い。
  zou-no *hana-ga* *nagai*
  elephant-POSS nose-SUBJ long
  The nose of an elephant is long.

  (6) 長い 鼻
  nagai *hana*
  long nose
  a long nose

- na-adjectives for the most part belong to the Sino-Japanese stratum, i.e. the class of old Chinese loan words, but also comprise e.g. loan words from western languages. In attributive position they take the suffix -na; in predicative position they need to be accompanied by a copula, which makes them similar to nouns. For this reason, they are also called noun-like adjectives.

  (7) この-家-は 立派 だ。
  kono-ie-wa *rippa* *da*
  this-house-TOP fine COP
  This house is fine.

  (8) 立派-な 家
  rippa-na *ie*
  fine house
  a fine house

Japanese nouns are not inflected for number or gender. Their grammatical function within the sentence is indicated by means of so-called particles, which are added as suffixes to the respective noun:

(9) 私-は 友達-に 本-を あげました。
watashi-wa *tomodachi-ni* *hon-o* *agemashita*
I-TOP friend-IOBJ book-DOBJ gave
I gave the book to the friend.

Japanese is a strongly pro-drop language in that any argument of the predicate, including subject and direct object, can be left out if they are clear from the context:

(10) あげました。
agemashita
gave
[I] gave [it] [to him].

Japanese is classified as a subject-object-verb language and as a head-last language, i.e. the modifying element always precedes the modified element:

(11) 広い 部屋
hiroi *heya*
large room
a large room

(12) 弟-の 部屋
otouto-no *heya*
my.little.brother-POSS room
my little brother's room

(13) 母-が 片づけた 部屋
haha-ga *katazuketa* *heya*
my.mother-SUBJ tidied.up room
the room that my mother tidied up

(Ebi and Eschbach-Szabo 2015, p. 133)

Relative clauses always directly precede their head, as shown in example 13. This makes it very easy to automatically determine the head of a relative clause, in contrast to some cases e.g. in English:

(14) *The wife* of Barack Obama, whose name is Michelle Obama,...

(15) The wife of *Barack Obama*, who is former president of the United States,...

However, since there are no relative pronouns in Japanese, it can be difficult to automatically figure out which grammatical role the head has within the relative clause:

(16) 噛みついた　犬
kametsuita　*inu*
bit　dog
the dog that bit [someone/something]/ the dog that [some other animal] bit

Certain heuristics can help in figuring out the correct grammatical role of the head in the majority of cases; for example, if the relative clause contains no overt subject, chances are high that the head serves as the subject of the relative clause.

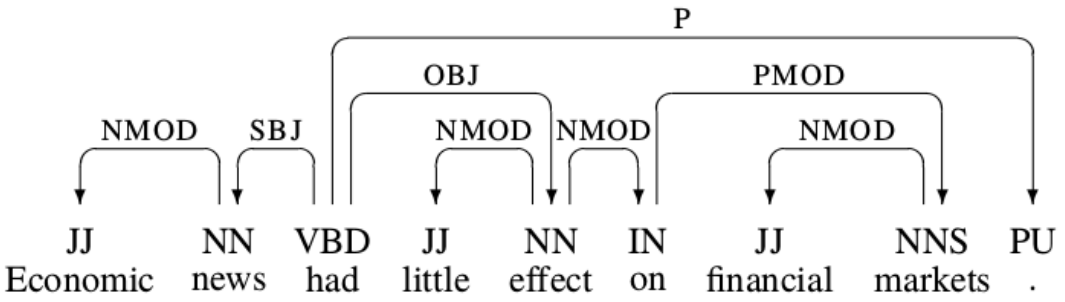


Figure 5: Dependency structure of an English sentence from the Penn treebank (Nivre 2005, p. 2)

2.7 DEPENDENCY GRAMMAR AND DEPENDENCY PARSING

Dependency grammar has a long history in descriptive linguistics, but played only a rather marginal role both in theoretical and computational linguistics until fairly recently (Kübler et al. 2009). The modern tradition of dependency grammar started with the work of the French linguist Lucien Tesnière, which was published posthumously in the 1950s (Tesnière 1959). Since then, a number of different dependency grammar frameworks have developed, some of the most well-known and influential of which are the Prague School's Functional Generative Description (Sgall et al. 1986), Mel'čuk's Meaning-Text Theory (Mel'čuk 1988), Hudson's Word Grammar (Hudson 1990; 2007), Dependency Unification grammar (Hellwig 1986) and Lexicase (Starosta 2015).

The tradition of dependency grammar comprises a large and fairly diverse family of grammatical theories that share a common core of assumptions about the nature of syntactic structures. The most important of these is probably that an essential part of the syntactic structure of sentences consists of binary, asymmetrical relations holding between words, which are called dependency relations. Hence, one characteristic of many forms of dependency grammar is the lack of phrasal nodes, as compared to representations based on constituency, as can be seen by comparing figures 5 and 6. A dependency relation holds between a subordinate word, called here a dependent, and another word on which it depends, called among other things the head; depending on the specific framework at hand, other terms than *dependent* and *head* may be used. Again depending on the framework, the arrows may point from the head to the dependent, or the other way around. Each arrow is assigned a label, indicating the kind of

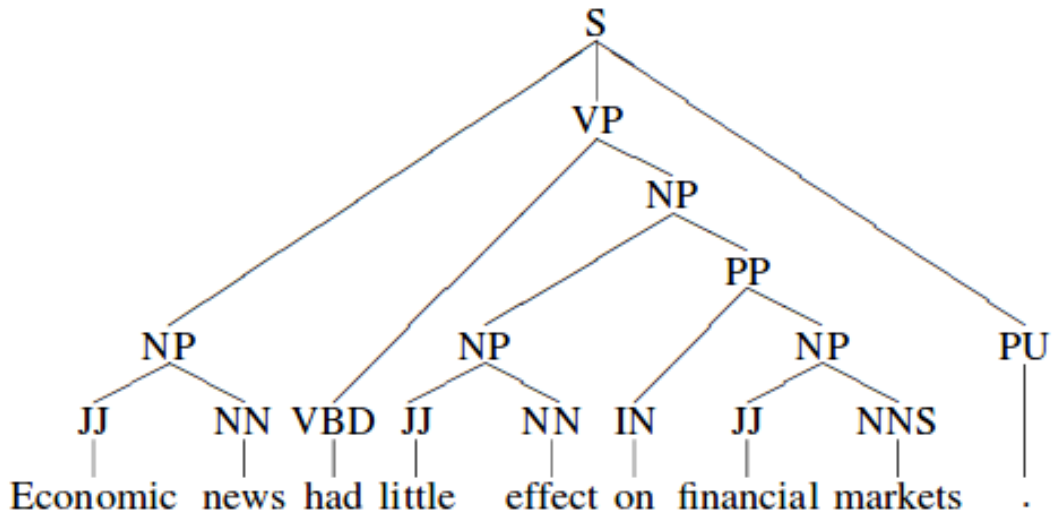


Figure 6: Constituent structure of an English sentence from the Penn treebank (Nivre 2005, p. 2)

dependency that holds between the words linked this way. For example, in figure 5 the noun *news* is the subject of the verb *had*, while the adjective *economic* is a modifier of *news*. The inventory of dependency types varies from one framework to the other: For example, traditional, more surface-oriented grammatical functions like *subject* or *object*, as shown in figure 5, or semantic role types such as *agent* or *patient* may be used. In principle, the dependency relations can also be left unlabeled; however, this is more common in practical parsing systems than in linguistic theories (Nivre 2005).

A dependency structure representation encodes information differently than a constituency or phrase structure grammar representation, which is the most widely used kind of syntactic representation both in theoretical linguistics and NLP: A dependency structure represents head-dependent relations between words, classified by functional categories such as subject or object, while a phrase structure represents the grouping of words into phrases, classified by structural categories such as noun phrase or verb phrase. However, the two approaches are not mutually exclusive or opposite, since the differences only concern what is explicitly encoded in the respective representation: Phrases can be distinguished in a dependency structure by letting each word represent a phrase consisting of the word itself and all the words that are transitively dependent upon it. Conversely, functional categories like subject or object can be identified in

a phrase structure in terms of structural configurations, such that e.g. the subject is identified as the NP under S. However, practical experience has shown that the automatic conversion from one type of representation to the other is a non-trivial task. Furthermore, while there is a core of syntactic constructions for which the analyses given by the different frameworks within dependency grammar agree in all important aspects, notably predicate-argument and head-modifier constructions, there are also a number of constructions for which there is no clear consensus about whether they should be regarded as head-dependent relations in the first place, and if so, what should be regarded as the head and what should be considered the dependent. Most notably, this concerns coordination, which is problematic for most theoretical traditions, and constructions involving auxiliary verbs. As a result, there are important differences among different frameworks with respect to whether dependency analysis is assumed to be not only necessary, but also sufficient for the analysis of syntactic structure in natural language, and some frameworks allow for a restricted form of constituency analysis e.g. for constructions that would be difficult to account for in a pure dependency analysis (Kübler et al. 2009).

One important aspect of dependency grammar are the criteria for establishing dependency relations and for distinguishing between the head and the dependent in such relations. The criteria usually proposed in this context are a mix of syntactic (e.g. *The form of the dependent depends upon the head*) and semantic criteria (e.g. *The head determines the semantic category of the overall construction, the dependent gives a semantic specification*), and one may ask how dependency grammar deals with this inconsistency. One solution employed in some frameworks is to posit the existence of several layers of dependency structure, such as surface syntax and deep syntax, or to introduce several kinds of dependency relations, such as endocentric and exocentric relations.

Overall, in case of dependency parsing the connections between theoretical frameworks and computational systems are often indirect, probably more so than in case of theories and parsers based on constituency analysis, which may be due to the relatively lower degree of formalization of dependency grammar theories in general. Dependency parsing can be viewed as the problem of mapping an input sentence, consisting of a list of words, to its dependency graph. Automatic dependency parsing started with the work on context-free dependency parsing by Hays and Gaifman in the 1960s (Hays 1964, Gaifman 1965). Gaifman proved several equivalence results relating his dependency systems to context-free grammars; in particular, he showed that while any dependency system can be converted to a strongly equivalent context-free grammar, the inverse is only true for a restricted subset of context-free grammar. This finding has often

been quoted to explain the lack of interest in dependency parsing for the subsequent 25 years or so, since it led to the view that dependency grammar is only a restricted form of context-free grammar, and hence not of much interest. However, in recent years dependency parsing has attracted considerable interest and has been used in a number of different applications of NLP, such as information extraction (Wu and Weld 2010), machine translation (Quirk et al. 2005) or question answering (Wang et al. 2007). The reasons for this development are manifold (Kübler et al. 2009): For one, issues of limited generative capacity have lost some of their significance due to the increasing importance of other problems such as robustness. Dependency grammar is better suited for languages with a free or flexible word order than phrase structure grammar, making it possible to analyze typologically diverse languages within one common framework. Furthermore, certain relationships e.g. between head and modifier, or predicate-argument structures, which are directly encoded within a dependency tree, are especially useful for a number of different NLP tasks, such as machine translation or information extraction. In addition, dependency parsing can be implemented in a rather efficient fashion: The task of parsing is in some sense more straightforward, since the dependency tree contains one node per word, which makes it the parser's job to only connect existing nodes, not to postulate new ones as in phrase structure parsing. Furthermore, dependency parsing lends itself to a word-at-a-time operation, i.e. to parsing by accepting and attaching words one at a time instead of waiting for complete phrases.

Similarly to other NLP techniques, approaches to dependency parsing can be broadly classified into grammar-based and data-driven ones (Nivre 2005). The grammar-based approaches were introduced in the 1960s with the work by Hays and Gaifman described above. Again similarly to other fields of NLP, in recent years data-driven methods, which are based on machine learning from large sets of syntactically annotated sentences, have attracted the most attention; most methods that have been proposed in recent years fall into one of two categories (Kübler et al. 2009):

- Graph-based dependency parsing starts by defining a space of candidate dependency graphs for a given sentence. The learning problem corresponds to inducing a model for assigning scores to the candidate dependency graphs of a sentence; the parsing problem amounts to finding the highest-scoring dependency graph for a given input sentence, given the induced model. Graph-based methods of dependency parsing were introduced by Eisner (1996).

- Transition-based dependency parsing starts by defining a transition system, or state-machine, for mapping a sentence to its dependency graph. The learning problem in this case corresponds to the induction of a model for predicting the next state transition, given the transition history; the parsing problem amounts to the construction of an optimal transition sequence for the given input sentence. This approach is sometimes referred to as shift-reduce dependency parsing, since it is inspired by deterministic shift-reduce parsing for context-free grammars. The transition-based approach to dependency parsing was first explored by Kudo and Matsumoto (2002). Both the MaltParser (Nivre et al. 2006), which was used for dependency parsing the English and Spanish corpus in Walter (2017), and the J.DepP parser (Yoshinaga and Kitsuregawa 2014), which we will use for dependency parsing our Japanese corpus, fall under this paradigm.

3

RELATED WORK

3.1 CROWDSOURCING IN NLP

Crowdsourcing, which is one of the approaches to the generation of ontology lexicons this thesis will deal with, can be used for a variety of NLP-related tasks. Callison-Burch and Dredze (2010) provide an overview of 24 crowdsourcing experiments in the area of NLP carried out on Amazon's Mechanical Turk (AMT), and the kinds of tasks presented there fall into two broad categories, annotation-based and translation-based tasks. In their survey, annotation-based tasks include the creation of corpora for fields such as word sense disambiguation, textual entailment or knowledge extraction, the annotation of speech and vision data with linguistic information, the annotation of sentiment, polarity and bias, and annotation tasks related to information extraction and information retrieval. Most of the translation-based tasks Callison-Burch and Dredze (2010) report on deal with the creation of bilingual parallel corpora that can be used as training data for machine translation systems.

One of the first reports on using crowdsourcing for the retrieval of NLP data was by Snow et al. (2008). They wanted to find out if crowdsourcing would be a viable approach to reduce the cost and temporal effort of annotating language data. In order to test this, they used AMT to carry out five different natural language understanding tasks — affect recognition, word similarity, recognizing textual entailment, event temporal ordering and word sense disambiguation — which they felt would be natural and learnable enough for non-experts, and for which they had gold standard labels from expert labelers available. Furthermore, they restricted their study to tasks where only a multiple choice response or numeric input within a fixed range is required. One of their findings was that while, as one would expect, individual expert labelers outperform individual non-expert ones, a strong correlation between expert and non-expert annotators can be achieved by aggregating the judgments of multiple non-experts. As a proof of concept, for one of their crowdsourcing tasks — affect recognition — they showed that using non-expert labels for training machine learning algorithms can be as effective as using gold standard annotations, or can even outperform these. A possible explanation for the latter would be that individual labelers, including experts, tend to

have a strong bias, and the high diversity of annotators achieved through crowdsourcing may have the effect of reducing annotator bias and hence increase system performance. In addition, by collecting 21,000 labels for just over 25 dollars they showed that through crowdsourcing a large amount of data can be collected at a fraction of the usual cost. They conclude that many large language labeling tasks can be effectively designed and carried out as crowdsourcing tasks at a fraction of the usual expense.

An example of how annotation-based crowdsourcing tasks can help in the generation of lexical resources is provided by Mohammad and Turney (2013), who used AMT for an emotion annotation task that resulted in a lexicon (EmoLex[1]) with over 10,000 word-sense pairs, with each entry listing the association between the word-sense-pair and eight basic emotions, and providing annotations on whether the word-sense pair has a positive or negative semantic orientation.

As mentioned before, apart from the annotation of language data another large part of NLP-related crowdsourcing tasks is concerned with the translation of texts. Since our approach to crowdsourcing ontology lexicons is also translation-based, insights from these kinds of experiments proved to be very valuable for our own work. As an example, Zaidan and Callison-Burch (2011) conducted a crowdsourcing experiment in which they tried to create a bilingual sentence-aligned parallel corpus for statistical machine translation (SMT). While SMT has been shown to produce state-of-the-art results for language pairs for which there is ample data, large bilingual corpora exist only for relatively few language pairs and generating such corpora by conventional means, for example by hiring professional translators, is a very costly task. Therefore, they examine the idea of creating low cost translations via crowdsourcing. As a proof of concept they recreate an Urdu-to-English sentence-aligned parallel corpus that had already been generated by professional translators, which allows them to compare the quality of professional and non-professional translations. One important difference from labeling tasks such as those described above is that in case of a translation task no discrete set of possible labels is given, but rather a diverse and complex space of possible outputs is available. Hence, a different and possibly more elaborate approach to quality control is required. Zaidan and Callison-Burch (2011) propose a four-stage workflow design that involves three different crowdsourcing tasks — one for retrieving multiple translations per source sentence, one for editing these translations and one for ranking these edited translations according to their correctness — and a final stage in which the best edited translations are selected by means of a machine learning-inspired approach that assigns a score to each translation based on the rank assigned during the third

[1] http://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm

crowdsourcing task and other features such as the translation's length or the country of origin of the translator. Simplifying the workflow significantly affects the quality of the output, and it seems that in case of sentence-based translation simple aggregation measures are not helpful and such an elaborate quality control mechanism would be justified. Therefore, in our own work we also made use of a multiple-stage workflow with a further crowdsourcing stage that served to evaluate the results retrieved during the actual translation task. Similarly to what has been found by Snow et al. (2008), one insight from the paper is that while many of the individual non-expert translators produce low-quality, disfluent translations, which are much worse than professional translations and only slightly better than automatic translations, soliciting multiple translations and aggregating them by means of the workflow just described allows one to still get high-quality output. While the total cost is more than an order of magnitude lower than that of professional translation, it is still significantly higher than the costs reported for the labelling tasks by Snow et al. (2008): Zaidan and Callison-Burch (2011) paid under 1,500 dollars to collect 7,000 translated sentences (716.80$), with 17,000 edited translations (447.50$) and 35,000 rank labels (134.40$). The actual translation stage is significantly more expensive than the other stages, which is to be expected since it is also the most demanding and complex task out of the three, and which is also consistent with what we found out during our own crowdsourcing experiments. However, the costs of such a translation task also seem to significantly depend upon the language pair under consideration and in particular upon the size of the languages in terms of their number of speakers, as could be shown by (Ambati and Vogel 2010) who conducted similar crowdsourcing experiments on translation for a number of different language pairs.

Apart from sentence-based translation, crowdsourcing has also been used for word-based translation tasks. While in our crowdsourcing experiments we worked with sentence-based translations, in the end we were only interested in those parts of the translation that correspond to a candidate verbalization of the ontology element at hand; hence, in some respect our approach also bears similarities to word-based translation tasks. Irvine and Klementiev (2010) present results from using AMT to generate translation lexicons between English and a large set of under-resourced languages. As their starting point, they generate candidate translations for 100 English words in each of 42 foreign languages using Wikipedia and a lexicon induction framework as follows: They mine Wikipedia for articles in these 42 languages that have interlanguage links with English and then compare the foreign language articles to the corresponding English articles, drawing up lists of word pairs that are likely good translations of

each other by representing each word as a vector of contextual word indices, using a small seed dictionary to project the contextual vector of a given source word into the respective target language, and scoring its overlap with the contextual vectors of candidate translations. The idea behind this is that tokens which tend to appear in the context of a given type in one language should be similar to contextual tokens of its translation in the other language. The top scoring target language words obtained this way are then used as input to the AMT task: Crowdsourcing workers are shown word pairs consisting of an English seed word and one of the top scoring target language words, and are asked to provide information on whether the latter provides a good translation of the former. Hence, in this workflow crowdsourcing is only used in an evaluation step to check the quality of the generated lexicon. To ensure accuracy, three different workers check each word pair, and negative and positive control candidate translations are used to test whether the workers provide correct results; this is basically the same approach to quality control we chose in our own crowdsourcing experiments. The crowdsourced annotations are evaluated by adding positively annotated pairs into the seed dictionary used for the induction of candidate translations and comparing the output retrieved in the next loop of the lexicon induction framework by means of the extended seed lexicon against complete available dictionaries, which in general shows a significant increase in accuracy. One of the insights from their work is that crowdsourcing workers are more successful in annotating some languages than others, which is to be expected given that the workers are not evenly distributed around the world or among the world's languages. They showed that it is possible to create bilingual lexicons between English and 37 out of the 42 low-resource languages they experimented with. For the remaining five languages no workers could be found at all, and only for three languages — Hindi, Spanish and Russian, i.e. languages with a large number of speakers — the crowdsourcing task completed within at most twenty hours. Overall, they conclude that AMT is a valuable resource for gathering cheap annotations for most of the languages they explored, and that these annotations provide a useful feedback in building a larger, more accurate lexicon.

3.2 AUTOMATIC INDUCTION OF LEXICA

M-ATOLL's corpus-based approach makes use of corpora to extract verbalizations of ontology elements and information about the linguistic properties of these verbalizations, such as their part-of-speech or the subcategorization frames they take. Already long before M-ATOLL, corpora have been used to gather a wide variety of different types of lexical information. One field of particular interest e.g. for ontology building (Grigonyte 2010) is the extraction of semantic relationships between words, such as the following:

- The relation of synonymy is given if two words or phrases within the same language have (nearly) the same meaning. Cruse (1986) defines (propositional) synonymy in terms of truth conditional relations between parallel sentences in which two lexical items X and Y occupy identical structural positions:

  > X is a propositional synonym of Y if (i) X and Y are syntactically identical, and (ii) any grammatical declarative sentence S containing X has equivalent truth-conditions to another sentence S', which is identical to S except that X is replaced by Y.(Cruse 1986, p. 88)

  As an example, *fiddle* and *violin* would be propositional synonyms in this sense, as e.g. *He plays the violin very well* is true under exactly the same conditions as *He plays the fiddle very well*. Synonym detection and extraction is relevant to a large number of NLP applications such as Information Retrieval, Thesaurus Extraction or Anaphora Resolution, and hence a considerable amount of research exists in this field. Some of the earliest contributions to the field of automatic synonym extraction would be Amsler (1980) and Michiels and Noël (1982); more recent work in this field includes e.g. Ohshima and Tanaka (2009) and Dias and Moraliyski (2009).

- Antonyms, in a broad sense, are words with opposite meanings. For example, depending on the given sense, two possible antonyms of *light* would be *dark* and *heavy*. While intuitively the relation may be clear, giving an adequate linguistic definition of antonymy is rather difficult (Cruse 1986, p. 197-198). There seems to be significantly less work in NLP on antonyms than on synonyms; some contributions to automatic antonym detection and extraction include e.g. Lin et al. (2003), Turney (2008) and Wang et al. (2010), which describe systems for both synonym and antonym extraction.

- Hyponymy is a hierarchical relation that corresponds to the inclusion of one class in another. According to Cruse (1986, p. 88-89), a lexical item X can be said to be a hyponym of a lexical item Y — in which case Y would be the hypernym of X — if an expression of the form *A is* $f(X)$ entails but is not entailed by *A is* $f(Y)$. Hence, e.g. *dog* would be a hyponym of *animal*, since *This is a dog* entails *This is an animal*, but not vice versa. Detecting and extracting hyponym-hypernym relations can e.g. be useful for the creation of taxonomic hierarchies that can be used in ontology building. Similarly to synonymy, a substantial amount of research on automatic hyponymy detection exists; some examples are Hearst (1992), Snow et al. (2005), Cimiano et al. (2005), Igo and Riloff (2009) and Rei and Briscoe (2014).

Corpus-based methods for the extraction of semantic relationships can be broadly classified into pattern-based approaches and approaches based on distributional similarity. The former is the older class of strategies and was pioneered by Hearst (1992), who used specific lexico-syntactic patterns like `X and other Y` or `X such as Y` for the detection of hyponym-hypernym relations. Later work in this field often focused on extending these so-called Hearst patterns in an automatic way, for example by using web data as evidence (e.g. Pantel and Pennacchiotti (2006)). Furthermore, these lexico-syntactic patterns were also used to detect other types of semantic relations; for example, Maynard et al. (2009) identify the pattern `X (Y)`, which can be used for synonymy detection. In general, systems based on lexico-syntactic patterns exhibit very high precision, but rather low recall, which is related to the specificity, and rareness, of the employed patterns (Grigonyte 2010, p. 64). Approaches based on distributional similarity rely on the hypothesis that words which are similar in meaning occur in similar contexts (Rubenstein and Goodenough 1965) and make use of the word space model (Schütze 1993) for representing words and their semantic similarity to each other: Words are modeled as vectors that represent certain of their features, for instance, the frequency of co-occurrence with other words within a text corpus. Then, the similarity of words — or of their vectors, respectively — can be modeled by means of different distance or similarity measures, such as the cosine value between them (Patwardhan and Pedersen 2006). The word space model allows the use of various methods based on machine learning for the detection of semantic relationships. For example, Lin (1998), Faure and Nedellec (1999) and Yan et al. (2009) employ clustering techniques for detecting semantically similar words, while examples of supervised learning methods applied to the extraction of semantic relations would be Girju et al. (2006) and Zhang (2008).

Apart from semantic relationships between words, another kind of important lexical information that can be extracted from corpora is the subcategorization frame of verbs, nouns and adjectives. The name is derived from the idea that words with a particular set of semantic arguments form one category, and each such category forms subcategories that express these arguments by different syntactic means. For example, in English the category of verbs that apart from the agent take a theme and a recipient as their semantic arguments can be subcategorized into verbs that express these arguments as an object and a prepositional phrase, as in *He donated a large sum of money to the church*, and verbs that in addition allow for a double-object construction such as *He gave the church a large sum of money* (Manning and Schütze 1999, p. 271-272). Information on subcategorization frames is e.g. important for parsing, where it can help disambiguate the attachment of arguments and recover the correct predicate argument relations by a parser (Carroll et al. 1998), and NLP applications that rely on information on predicate-argument structures, such as machine translation or information extraction (Hajič et al. 2004, Surdeanu et al. 2003). Due to their importance, information on subcategorization frames is also included in ontology lexicons generated by M-ATOLL. Before subcategorization frames could be automatically extracted from corpora, parsers made use of hand-generated lists of subcategorization frames (e.g. de Marcken (1990)). One of the earliest works in the field of subcategorization frame acquisition from corpora was by Brent (1991), whose system takes an untagged text corpus as its input and generates a partial list of verbs included in the corpus and the subcategorization frames they occur in. Brent's system was able to detect five different subcategorization frames; as his test corpus he used a 2.6 million words corpus based on the Wall Street Journal, for which the system was able to observe 2258 orthographically distinct verbs. Brent's priority was accuracy in identifying subcategorization frames, not efficient use of the avilable text, arguing that the large amount of available text would make up for a possible inefficiency of his system. The system consists of three modules: A Verb Detection module that finds some occurrences of verbs based on the positions of pronouns and proper nouns in a given sentence, a Subcategorization Frame Detection module that finds some occurrences of the five subcategorization frames using a small finite-state grammar that describes a fragment of English that is most useful for recognizing subcategorization frames, and a Subcategorization Frame Decision module that determines whether a given verb is genuinely associated with a given frame, or whether its apparent occurrences within that frame are due to error. Consistent with Brent's priorities, the system exhibits a very high accuracy, but a rather low effectiveness in detecting verbs and occurrences of subcategorization frames. An example of a lexical

resource created through the acquisition of subcategorization frames from corpus data is provided by Korhonen et al. (2006), who produced a computational subcategorization lexicon (VALEX[2]), which includes subcategorization and frequency information for 6,397 English verbs that was acquired from five corpora and the web using the subcategorization acquisition system of Briscoe and Carroll (1997), which is capable of categorizing 163 verbal subcategorization frames and returning relative frequencies for each frame found for a verb. Evaluation against both a manual analysis of a part of the corpus data and two already existing, manually built subcategorization dictionaries — COMLEX (Grishman et al. 1994) and ANLT (Boguraev and Briscoe 1987) — shows that while the basic lexicon is very noisy, filtering e.g. based on the frequency by which verbs or subcategorization frames are encountered in the corpora can significantly improve the quality of the lexicon. While most systems for the acquistion of subcategorization frames focus on frames of verbs, Preiss et al. (2007) present a system that is also able to acquire subcategorization frames of nouns and adjectives from English corpus data and which can be used to acquire comprehensive subcategorization lexicons for verbs, nouns and adjectives for NLP purposes. The system incorporates a rule-based classifier which identifies 168 verbal, 37 adjectival and 31 nominal subcategorization frames from Grammatical Relations (GRs), i.e. head-dependent relations such as `subj` or `obj`, in contrast to the constituency-based syntactic parse trees most other approaches rely on. Hence, in a way the approach employed here is very similar to that used in M-ATOLL's corpus-based approach, where the subcategorization frame of a candidate verbalization occurrence is determined by the dependency pattern in which it occurs. According to Preiss et al. (2007, p. 2), "dependency relationships which the GRs embody correspond closely to the head-complement structure which subcategorization acquisition attempts to recover, which makes GRs ideal input to the S[ub]C[ategorization]F[rame] classifier". In the first step, these head-dependent relations are extracted from the corpus data by means of a GR parser. Then, the GR sets obtained for each sentence are fed to the rule-based classifier which matches them with the corresponding subcategorization frames. The frames recognized by the classifier were obtained by manually merging the frames exemplified in the COMLEX, ANLT and NOMLEX (Macleod et al. 1997) dictionaries and adding further frames found by manual inspection of unclassifiable examples during the development of the classifier. The output of the classifier is then used to construct lexical entries for each combination of a word and subcategorization frame encountered, with additional information such as the raw and relative frequency of the subcategorization frame with the word in question being provided. Finally, the

[2] `https://www.ilexir.co.uk/valex/index.html`

retrieved entries are filtered based on empirically determined thresholds on the relative frequencies of subcategorization frames in order to obtain a more accurate lexicon. For evaluation purposes, Preiss et al. (2007) generated lexicon entries for 183 verbs, 30 nouns and 30 adjectives from the BNC and compared these entries to a gold standard manually created from a subset of the test corpus. Furthermore, for the verbs they compared the performance of their system against another system for the acquisition of subcategorization frames of verbs (Briscoe and Carroll 1997). Their system achieves better performance on all measures, but especially on precision. For nouns and adjectives, the system achieves very high precision, but rather low recall relative to the gold standard, and many occurrences of subcategorization frames available in the gold standard were not detected by the classifier. According to the authors, the main reason would be that the GR parser often fails to select the correct analysis for occurrences of adjectives and nouns, and the problem could be alleviated by passing the top n-ranked parses returned by the parser, instead of the single highest ranked parse, as input to the classifier.

3.3 LEXICALIZATION OF ONTOLOGIES

As an example of an approach to ontology lexicalization different from the one taken by M-ATOLL, Pazienza and Stellato (2006) present a framework that facilitates the manual linguistic enrichment of ontologies and which led to the development of Ontoling[3], a plugin for the ontology devlopment tool Protégé[4]. The basic idea behind Ontoling is that ontology lexicalization should be part of the ontology development process, and ontology development tools should reflect this need and support users with specific interfaces for browsing linguistic resources such as lexical databases or terminologies. One main challenge in developing such an interface is that linguistic resources considerably vary in format. In order to deal with this, Pazienza and Stellato (2006) developed a Java library that is able to provide a uniform interface to a variety of diverse and heterogeneous linguistic resources. The GUI of Ontoling consists on the one hand of a Linguistic Resource browser, which is responsible for letting the user explore the loaded linguistic resource, and an Ontology Enrichment panel, which offers a view of the ontological data in the classic Protégé style. On the one hand, Ontoling allows the user to add further labels to an ontology element — realized through `rdfs:label` in case of OWL-based ontologies — or to change the name of an element to a term from the linguistic resource. It is mentioned that reifying linguistic terms as concrete ontological elements, instead of treating them as labels attached to concepts, would be advantageous and that this is planned for the future. In any case, Ontoling treats linguistic information on the elements of an ontology as an element of the ontology itself, and does not store this kind of information in a separate resource such as an ontology lexicon. On the other hand, in case a taxonomical linguistic resource is loaded, it is possible to explore the hyponymy relations contained in it and reproduce the respective trees within the ontology under development. Hence, Ontoling not only allows for the inguistic enrichment of ontologies, but can also be helpful in creating new ontologies or improving existing ones.

Marginean and Eniko (2016) propose a method for the extraction of potential lexical expressions for DBpedia properties from a Wikipedia-based text corpus, which makes their basic strategy very similar to that behind M-ATOLL's corpus-based approach. However, while M-ATOLL works with a dependency-parsed corpus and uses a set of language-specific dependency patterns for the extraction of candidate lexicalizations, the system presented by Marginean and Eniko (2016) uses a corpus annotated with

[3] `http://art.uniroma2.it/software/OntoLing/`
[4] `http://protege.stanford.edu/`

Semantic Role Labeling (SRL) and unsupervised learning for the identification of verbalizations relevant to a property. Their approach comprises three main steps: The Data Gathering Module serves to extract triples relevant to the property at hand from DBpedia's triple store, followed by the extraction of Wikipedia articles describing the resources from these triples: First, similar to M-ATOLL's corpus-based approach, the system retrieves all triples from DBpedia's endpoint which are connected by the respective property, i.e. which have the form

```
entity1 property entity2 .
```

Then, the module queries Wikipedia for the article corresponding to the first entity; the collection of articles collected this way for all relevant triples are then passed as input to the Data Processing Module. In this step, the goal is to extract the sentences most probably related to the property from the articles and to build SRL-annotated trees from them. First, from each Wikipedia article selected due to a triple of the form given above, the module selects only those sentences for further processing in which both `entity1` and `entity2` are mentioned. Depending on the property under consideration, the sentences either need to contain the complete labels of both entities, only one word from each entity's label, or lexical items that are coreferent with the labels, as determined by the coreference resolution module of StanfordCore NLP[5]. After this filtering step, the extracted sentences are annotated using Semantic Role Labeling, which detects the predicates in a given sentence together with their arguments and assigns labels that indicate which roles these arguments occupy. For each identified predicate there is a corresponding tree having as its root node the predicate, and the predicate's arguments as its leaves. Since each sentence may contain several predicates, a sentence can be represented as a graph of trees. The set of SRL-annotated trees generated from the filtered sentences forms the input of the final Unsupervised Learning Module, which builds clusters of candidate verbalizations using the Spectral Clustering algorithm. Spectral Clustering takes as its input a similarity matrix, which is built by computing the pairwise similarity of the obtained SRL trees according to three metrics which are based on the value of the root element (i.e. predicate) of the SRL trees, the number of common role labels in both trees, and the number of common lexical items in role position, respectively. The weighted sum of these metrics serves as the similarity value of two trees, and the resulting similarity matrix is used as input for the Spectral Clustering algorithm, whose output consists of a set of clusters of SRL trees. A subset of these custers are selected as the final output according to a two-step

[5] `https://stanfordnlp.github.io/CoreNLP/`

filtering method: First, all clusters are filtered out in which more than two different lexical items occur in the root node positions of the contained SRL trees. After that, the remaining clusters are further filtered either based on the semantic similarity of the value of the root node of the trees to the property at hand, which is computed according to a word space model obtained from GoogleNews, or based on whether the entities due to which the sentences were extracted for further processing are mentioned in the SRL trees contained in the cluster. The actual set of candidate verbalizations is then generated from the predicates located in the root node position of the SRL trees. Marginean and Eniko (2016) compare their results for eleven properties to the training data for the CLEF 2013 ontology lexicalization task: While the lexicon they generated seems to contain a certain amount of noise, recall seems to be good.

4

CROWDSOURCING ONTOLOGY LEXICONS

4.1 OVERVIEW

While crowdsourcing has already been used for a number of different tasks related both to natural language processing and ontologies (Snow et al. 2008, Ambati and Vogel 2010, Acosta et al. 2013), to our knowledge prior to Lanser et al. (2016) there existed no reports on using crowdsourcing specifically for ontology lexicalization. Hence, whether ontology lexicons of good quality could be generated this way at acceptable costs was an open question. In the following, we will present an approach to generating a Japanese ontology lexicon for DBpedia by means of crowdsourcing, which can also be applied both to other languages and other ontologies. We used CrowdFlower to create a small Japanese ontology lexicon for ten exemplary elements from DBpedia's ontology according to a two-stage workflow, whose main underlying idea was to turn the ontology lexicalization task into a translation task. As a starting point for the translation we used a manually created English lexicon for DBpedia (Unger et al. 2013). The quality of the crowdsourced ontology lexicon, and alternatives to the workflow we employed in the first test run, will be discussed in chapter 6.

4.2 METHODOLOGY

4.2.1 *Overall Workflow*

A particular challenge when trying to crowdsource ontology lexicons is finding a task design that is suitable for workers with no knowledge about ontologies or lexical resources: Obviously, it would not make much sense to simply present the workers with an ontology element and then asking them to come up with a good verbalization, or even a whole lexicon entry. Therefore, we instead turn the lexicon generation task into a translation task. The starting point is a seed lexicon in English, in our case a manually created English lexicon for DBpedia (Unger et al. 2013) with over a thousand entries. We ask Japanese crowdsourcing workers to provide Japanese translations of the English verbalizations, with each Japanese translation being understood as a potential verbalization of the ontology element linked to the original English verbalization. As an example, let us assume we are looking for a Japanese verbalization of the property `author`, which in the DBpedia ontology links the classes `Writer` and `Book`. We would search the seed lexicon for entries which reference this property, and possibly among others, we would find an entry containing the verbalization *to write*. Our strategy would then be to ask the crowdsourcing workers for a Japanese translation of the verb *to write*, and each such translation would be treated as a candidate verbalization of `author` for our Japanese ontology lexicon. Figure 7 provides a visualization of this very general idea behind our workflow.

There may be English seed verbalizations which have more than one possible meaning, or which could be used to verbalize more than one ontology element. For example, *to write* cannot only be used to verbalize the relationship holding between an author and a book, but may link authors to all kinds of written work, and there may be target languages in which these different kinds of relationships are verbalized in different ways. Therefore, we need to ensure that the English verbalizations are understood in the right sense, which can be accomplished by presenting them to the workers embedded in some kind of context. We decided to present the English verbalizations within short sentences, which we automatically generated using the Lemonade tool (Rico and Unger 2015). Each such sentence is built on the one hand from the English seed verbalization currently looked at and on the other hand from a triple found in the DBpedia dataset that contains the associated ontology element. The way the dataset gets searched for a suitable triple depends on the type of ontology element we are dealing with:

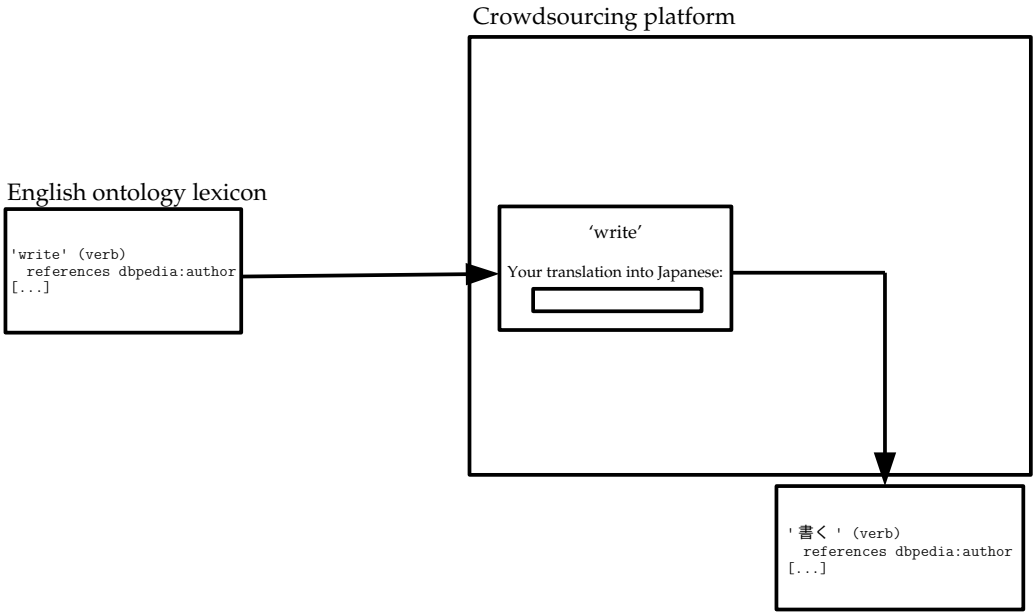


Figure 7: Basic structure of our crowdsourcing workflow

- When looking for a verbalization of a property, such as `author`, we would search the DBpedia dataset for a triple that contains the respective property as its predicate and include the subject and object of this triple in our automatically generated sentence along with the English seed verbalization. Hence, if in our example we retrieved the triple

  ```
  Don_Quixote author Miguel_de_Cervantes
  ```

  from the DBpedia dataset, the generated sentence would have the form *Miguel de Cervantes wrote Don Quixote*.

- In contrast, if we were looking for a verbalization of a class, such as e.g. `Book`, we would search the dataset for a triple that links some individual to the respective class by means of the property `rdf:type`, and would use that individual in the generated sentence together with the respective English seed verbalization. Hence, if for the class `Book` we had picked a triple of the form

  ```
  Don_Quixote rdf:type Book
  ```

  and the seed lexicon contained the verbalization *book* for the class `Book`, we would generate a sentence of the form *Don Quixote is a book*.

While using real-life sentences from a corpus would have also been an option, we decided against this, as in many cases such sentences tend to be rather long and not always clearly disambiguate the verbalization. In contrast, as they always reference entities that are conceptually linked to the ontology element the verbalization is meant to represent, our automatically generated sentences have a high chance of presenting verbalizations in an unambiguous way. Furthermore, the simple structure of these sentences not only makes the translation task easier for the crowdsourcing workers, but will probably allow us to eventually extract the Japanese verbalizations from the translated sentences automatically by means of the M-ATOLL framework.
One obvious challenge with a crowdsourcing task such as this one, where more than one correct answer may exist for a given piece of input data and there is no straightforward way to automatically check the validity of the answers provided by the workers, is quality control. We adopt an approach that is commonly used in translation-related crowdsourcing tasks (Zaidan and Callison-Burch 2011, Benjamin and Radetzky 2014) and which involves soliciting multiple translations per English sentence from distinct workers, plus a second crowdsourcing stage in which Japanese workers will be asked to evaluate the translations received in the first stage. Based on these evaluations we can

then decide which translations most probably include commonly accepted Japanese verbalizations of ontology elements that should be included in a Japanese ontology lexicon for DBpedia. As an example, assume we have shown the sentence *Miguel de Cervantes wrote Don Quixote* to three separate workers during the translation stage. Worker number one translated the sentence as given in example 17, while the other two only entered gibberish.

(17) ミゲル・デ・セルバンテス-は　ドン・キホーテ-を　書いた。
migeru de serubantesu-wa　*don kihoote-o*　*kaita*
Miguel.de.Cervantes-TOP　Don.Quixote-DOBJ　wrote
Miguel de Cervantes wrote Don Quixote.

Distinguishing between the useful answer and the gibberish ones automatically may be extremely difficult, or even impossible. However, during the following evaluation phase filtering out workers who do not provide acceptable evaluations of the translations automatically would be much easier and could e.g. be done through test sentences for which we already know in advance which translations are correct and which are not. Therefore, if in the evaluation stage we received the information that the sentence provided by worker number one is a good translation, while the other two are not, we could be rather confident that this information is correct.

Based on the findings from the evaluation stage, we would then decide that the translation given in example 17 most probably contains a valid Japanese verbalization of the property `author`. We would then have to extract this verbalization — the verb 書く (*kaku*) — from the translation and turn it into a complete lexicon entry with further information, e.g. about its part of speech. As mentioned before, in the future it should be possible to do this automatically by means of M-ATOLL; at the time of our crowdsourcing experiments, however, we still needed to perform these steps manually. The overall workflow of our approach is shown in Figure 8.

With a translation-related task such as this one, one obvious question is what potential advantages crowdsourcing the task would have over simply hiring a professional translator (Zaidan and Callison-Burch 2011). On the one hand, the latter approach would probably be very expensive, and one may assume that crowdsourcing the task will be considerably cheaper. On the other hand, with input from only one person, the variance of the received verbalizations would probably be lower than if potentially a lot of different people provide translations.

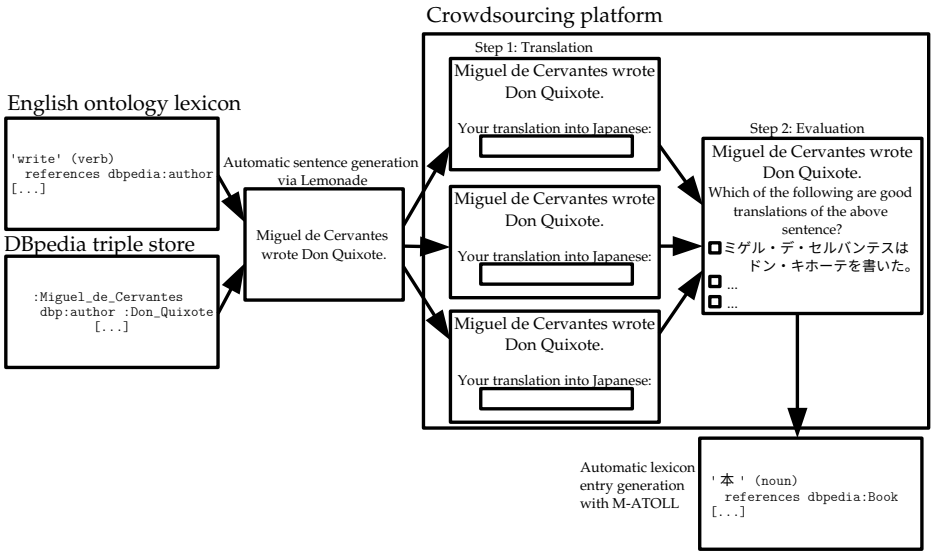


Figure 8: Workflow of our approach for the generation of a Japanese ontology lexicon

4.2.2 *Choice of Crowdsourcing Platform*

A lot of crowdsourcing-related research is carried out via Amazon's MechanicalTurk platform. However, as on Mechanical Turk only workers who are based either in the US or India can be paid in cash[1], well over 90 percent of workers come from either of these countries[2], which would make finding a sufficient amount of workers with a native language other than English or an Indian language rather difficult. Therefore, MechanicalTurk was not a good option for us. We also looked at a number of Japan-based crowdsourcing platforms; however, many of these seemed not very suitable for the crowdsourcing tasks we had in mind, either. On the one hand, some of them are based on an idea of crowdsourcing quite different from the kind of task we wanted to carry out: On some platforms like Lancers[3], companies can find freelancers for rather complex tasks, like designing a corporate design for them, while other platforms like Coconala[4] focus on tasks such as personal consulting. On the other hand, there are a number of platforms where only very specific types of tasks can be carried out, such as e.g. video production on Viibar[5].

We finally chose to work with CrowdFlower, which is another global crowdsourcing platform similar to MechanicalTurk. However, in contrast to the latter, CrowdFlower does not have any country-related restrictions on how workers can be paid, and so it seems likely that there is a higher diversity with respect to the countries of origin of the workers. This assumption seems to be backed by a survey done by CrowdFlower itself based on responses of workers to a questionnaire, where only 30% in total indicated either the US or India as their country of origin[6]. Furthermore, CrowdFlower's worker interface is available in Japanese, and — again in contrast to Mechanical Turk — workers for a task can be chosen according to their geographical location and their language skills.

However, the platform also has a number of characteristics that are rather disadvantageous with regard to our specific task at hand: First of all, requesters on CrowdFlower need to pay for every result submitted by a worker, no matter this result's quality. This is in contrast to what seems to be common practice on most other crowdsourcing platforms, where employers are able to look at and evaluate the received results and

[1] https://www.mturk.com/mturk/help?helpPage=worker#how_paid
[2] http://demographics.mturk-tracker.com/#/countries/all
[3] http://www.lancers.jp/
[4] http://coconala.com/
[5] http://viibar.com/
[6] https://success.crowdflower.com/hc/en-us/articles/202703345-Crowd-Demographics

decide which of these they actually want to use and pay for. As an incentive to still provide quality work, workers at CrowdFlower are ranked by the platform according to their performance on test questions, which are interspersed among the actual task and which are pre-labeled with known answers provided by the requester. Requesters can choose which minimum quality rank workers need to have in order to work on their task, and usually tasks offered only to workers of higher ranks pay better and are more interesting. Obviously, this kind of quality control only works for jobs where test questions can be formulated in the first place, i.e. where at least for certain input data a closed and predictable set of correct results exists. Hence, it is not suitable for translation tasks, as in most cases predicting all possible correct translations to a given seed sentence is simply impossible. We therefore had to make use of a number of alternative quality control mechanisms for the translation task, which will be described in the following.

4.2.3 *Quality Control*

As for a task such as ours one cannot really formulate test questions, which form CrowdFlower's main mechanism of quality control, we made use of a number of alternative control mechanisms that are commonly used with translation-related tasks (Irvine and Klementiev 2010, Zaidan and Callison-Burch 2011): Generally, a good strategy for discouraging people from cheating is to design one's task in a way that makes cheating as laborious and time-consuming as actually working on the job at hand. Therefore, we showed the English seed sentences — and, in case of the later evaluation stage, the Japanese candidate translations — to the workers in the form of images rather than text, so as to make it more difficult for people to make use of automatic translation services. This measure serves to prevent workers from using automatic translations services, but it does not help against workers who either just insert gibberish or who sincerely try to work on the task but either did not understand the instructions or for some other reason produce incorrect results. Detecting faulty output of this kind is the main purpose of the second stage of our workflow, in which workers are shown an English seed sentence together with its Japanese candidate translations and are asked to judge the quality of the latter. As mentioned before, for this second kind of crowdsourcing task, one can actually formulate test questions and hence filter out workers who do not submit reliable evaluations automatically. More details about how test questions work on CrowdFlower and how we generated those

questions for our task can be found in Section 4.3.3.
Finally, a further important measure of quality control concerns the choice of workers: CrowdFlower classifies workers into three groups according to their previous performance on test questions, and we only allowed workers of the highest quality group to work on our tasks, which is what CrowdFlower advises for tasks one cannot formulate test questions for[7]. Furthermore, for the translation task we experimented with different settings for the country of origin and language skills of allowed workers, as will be described in more detail in Section 4.3.2

[7] https://success.crowdflower.com/hc/en-us/articles/201855969-Guide-To-Running-Surveys

4.3 TEST STAGE

4.3.1 *Overview*

In order to gain some first experiences with crowdsourcing and to check whether our approach is feasible, we first conducted a test stage in which only for a small number of ontology elements Japanese verbalizations should be found: We started with ten ontology elements, the types of which were chosen so as to roughly mirror the overall distribution of element types (classes, object properties, datatype properties) in the DBpedia ontology. Furthermore, for each element type we chose one element with many (≥ 4) and one with some (2–3) verbalizations in the seed lexicon, plus one or two elements with only one verbalization. The actual ontology elements and their verbalizations from the English seed lexicon can be found in Table 2. We conducted several runs of both the translation and evaluation stage with different settings for

- the language of the task instructions (English or Japanese),
- the allowed countries of origin of the workers (no restrictions or Japan),
- and the language skills of the workers, as tested by a language proficiency test provided by CrowdFlower (no restrictions or Japanese)

4.3.2 *Translation Stage*

For each of the 25 verbalizations shown in Table 2 we automatically generated three example sentences in the manner described in section 4.2.1, and for each such sentence we asked for translations by three separate workers, totalling $3 * 3 * 25 = 225$ data rows to be retrieved. Figure 9 shows one such example task as it was presented to the workers, together with the English instructions we used. As mentioned above, we experimented with different settings for the countries of origin and language skills of allowed workers, and also tried out whether the language the task instructions are given in has any effect on the workers' performance.
One of the challenges of crowdsourcing is finding the right amount of payment for the workers that on the one hand provides an incentive for people to work on the task but on the other hand does not make it too attractive for cheaters. As we were not sure what kind of payment would be appropriate, we always started at a payment of one cent per sentence, and slightly increased the payment every time no one had worked on

Element type	URI	Verbalizations
Classes	`PowerStation`	generating station
		power station
		power plant
		generating plant
		electricity station
	`Star`	star
		sun
	`Artist`	artist
Object properties	`parent`	child
		father
		daughter
		son
		parent
		mother
	`occupation`	occupation
		to work
	`colourName`	color
Datatype properties	`numberOfStudents`	student population
		to have an enrollment of
		enrollment
		to serve
	`weight`	to weigh
		weight
	`budget`	budget
	`yearOfConstruction`	constructed

Table 2: Exemplary ontology elements and verbalizations used in the test stage

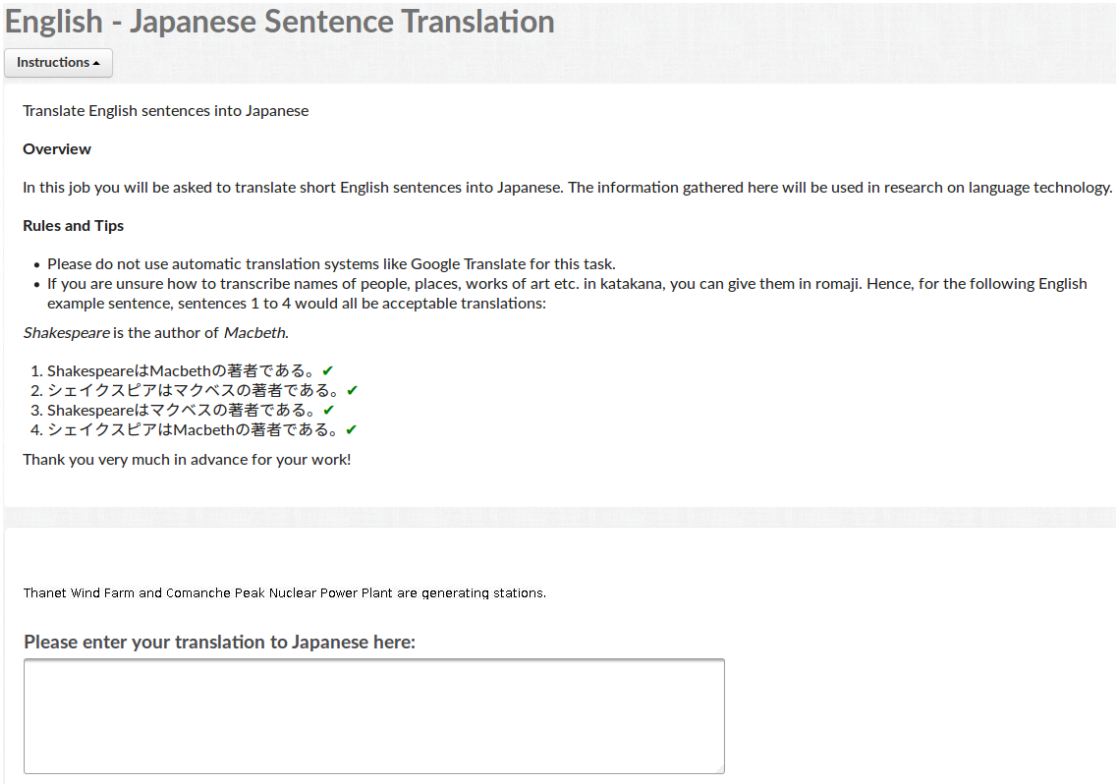


Figure 9: Exemplary task from the translation stage of our test run, together with the English instructions we used.

the task for a longer period of time (at least 24 hours). In order to estimate the quality of the translations, we randomly picked five translations from each worker and looked at whether they contained obvious semantic or grammatical errors.

In our first run, we only required the allowed workers to have passed CrowdFlower's Japanese proficiency test, but did not impose any restrictions on the workers' country of origin. Furthermore, the task instructions were given in English, as workers should not only have a good understanding of Japanese but also at least basic English skills for this task in order to understand the English seed sentences; CrowdFlower does not provide any English proficiency test. This run was finished very fast within only four hours, so we did not have to raise the payment of one cent per sentence. However, the overall results we retrieved in this run were of very poor quality, as there were many contributors who either only entered gibberish or gave grammatically incorrect word-by-word translations of the English sentences. A possible explanation would be that many people may cheat on CrowdFlower's language proficiency tests and pass them even though they do not speak the respective language at all: As mentioned before, the English sentences are presented to the workers as images, so for someone with a sufficient knowledge of both Japanese and English simply translating a sentence themselves should actually be less work than presumably entering it into some automated translation system by hand, or even looking up every single word on its own.

As a result, we did a second run of the translation task, which only differed from the first one in that now only workers from Japan were allowed to work on it. This time, results were significantly better, and there were no obvious word-by-word translations or workers who entered gibberish. However, at around one month this run also took much longer to finish. As can be seen in Table 3, while we already achieved a completion of nearly fifty percent at a payment of only one cent per sentence, in the end we had to raise payment to up to eight cent per sentence in order to also receive results for the last pending microtasks. It should be noted here that each worker was only allowed to provide translations for at most half of all English seed sentences. This setting was chosen so as to achieve higher variance, as otherwise it may have been possible for only three separate workers to complete the whole task. Working with a different setting here may of course have resulted in the task getting completed in a shorter time and at a lower maximum cost, as people who were satisfied with a lower payment per sentence may have worked on a larger number of sentences then.

Out of the three workers who delivered clearly low-quality results, two worked on the task only after payment per sentence had been raised to eight cent. While the low

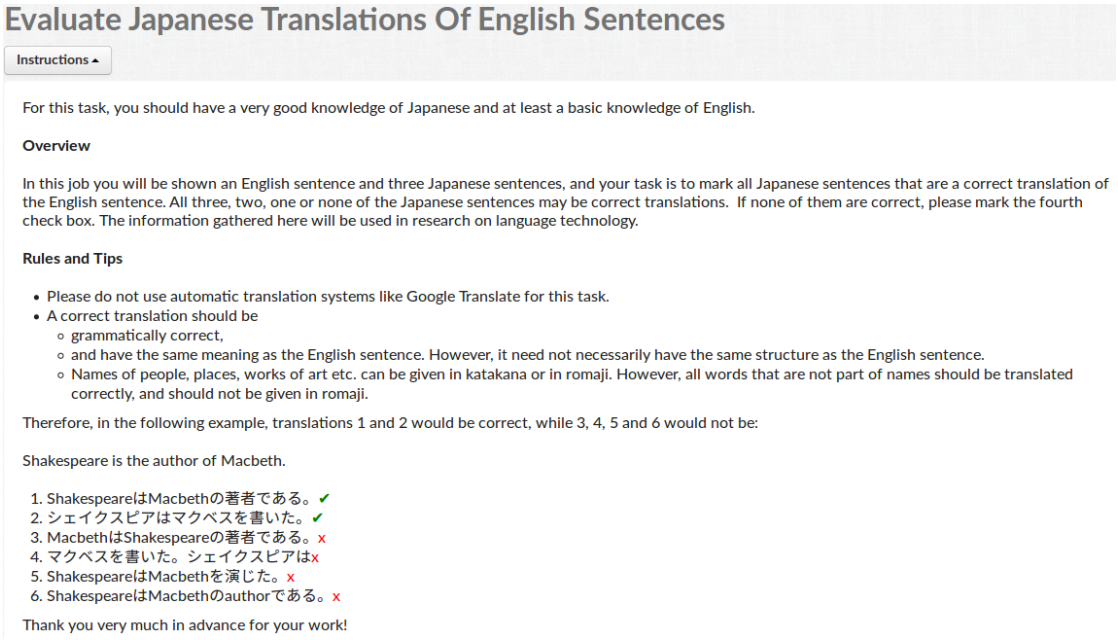
Cent/sentence	Total number of workers	Workers with low-quality contributions	Tasks completed
1	4	0	48.89%
4	6	1	86.67%
6	6	1	88%
7	7	1	93.33%
8	10	3	100%

Table 3: Results from second run of translation stage

overall number of workers does not allow one to make any definite statements here, this may be seen as a sign that at around this amount of payment there is a threshold at which this kind of task also becomes attractive for cheaters.
We wanted to know if the language the task instructions are given in has any significant effect on the quality of the results; therefore, we started a third run which only differed from the second one in that the instructions were now given in Japanese. However, we stopped this final run at around 72 percent completion, as it turned out that all workers who had contributed to this run so far had also contributed to the second one, delivering basically the same quality, and we assumed that also for the remaining microtasks the results would most probably not differ that much from those from the second run. In the following steps, we only considered the data from the second run.

4.3.3 *Evaluation Stage*

When we wanted to start the evaluation stage of our test run, it turned out CrowdFlower had removed Japan from the list of countries that can be used to filter which workers are allowed to work on one's task, and it had also deactivated the option to narrow down allowed workers to only those who speak Japanese. When we contacted CrowdFlower's support about this, we were told they had removed these options due to the small number of Japanese workers currently available through the site, and that they may get activated again some time in the future should CrowdFlower find a way to provide a larger Japanese workforce to the requesters who use their platform. This change means that we may not be able to use CrowdFlower for future crowdsourcing experiments. However, we decided to at least try to finish our current test run on the platform.
In each microtask of this stage the workers were shown one English seed sentence



Evaluate Japanese Translations Of English Sentences

Instructions ▲

For this task, you should have a very good knowledge of Japanese and at least a basic knowledge of English.

Overview

In this job you will be shown an English sentence and three Japanese sentences, and your task is to mark all Japanese sentences that are a correct translation of the English sentence. All three, two, one or none of the Japanese sentences may be correct translations. If none of them are correct, please mark the fourth check box. The information gathered here will be used in research on language technology.

Rules and Tips

- Please do not use automatic translation systems like Google Translate for this task.
- A correct translation should be
  - grammatically correct,
  - and have the same meaning as the English sentence. However, it need not necessarily have the same structure as the English sentence.
  - Names of people, places, works of art etc. can be given in katakana or in romaji. However, all words that are not part of names should be translated correctly, and should not be given in romaji.

Therefore, in the following example, translations 1 and 2 would be correct, while 3, 4, 5 and 6 would not be:

Shakespeare is the author of Macbeth.

1. ShakespeareはMacbethの著者である。✔
2. シェイクスピアはマクベスを書いた。✔
3. MacbethはShakespeareの著者である。x
4. マクベスを書いた。シェイクスピアはx
5. ShakespeareはMacbethを演じた。x
6. ShakespeareはMacbethのauthorである。x

Thank you very much in advance for your work!

Figure 10: English instructions we used in the evaluation stage of our test run

together with the three translations we had received for it in the translation stage, and they were asked to mark all translations that they considered correct. Figure 10 shows the English instructions we used for this stage, while figure 11 shows an exemplary evaluation task. For each seed sentence, we elicited evaluations of its translations by three separate workers; as a result, the number of microtasks in this stage was the same as in the preceding stage (225). In addition, we uploaded twenty test questions. In CrowdFlower these need to have the same structure as the actual microtasks; hence, each such test question consisted of an English sentence and three Japanese sentences for which we knew in advance which of them constituted correct translations of the English sentence and which did not, and that we pre-labeled accordingly. On the one hand, CrowdFlower uses these questions to test workers before the actual task starts in so-called quiz mode, where workers need to answer a certain amount of test questions — five in our case — before they can actually work on the task. On the other hand, also during the task itself a certain amount of the microtasks shown to the workers — twenty percent in our setup — are actually test questions. Workers need to answer a certain percentage of test questions correctly throughout the job — eighty percent in our case — or else CrowdFlower will keep them from working on further microtasks.

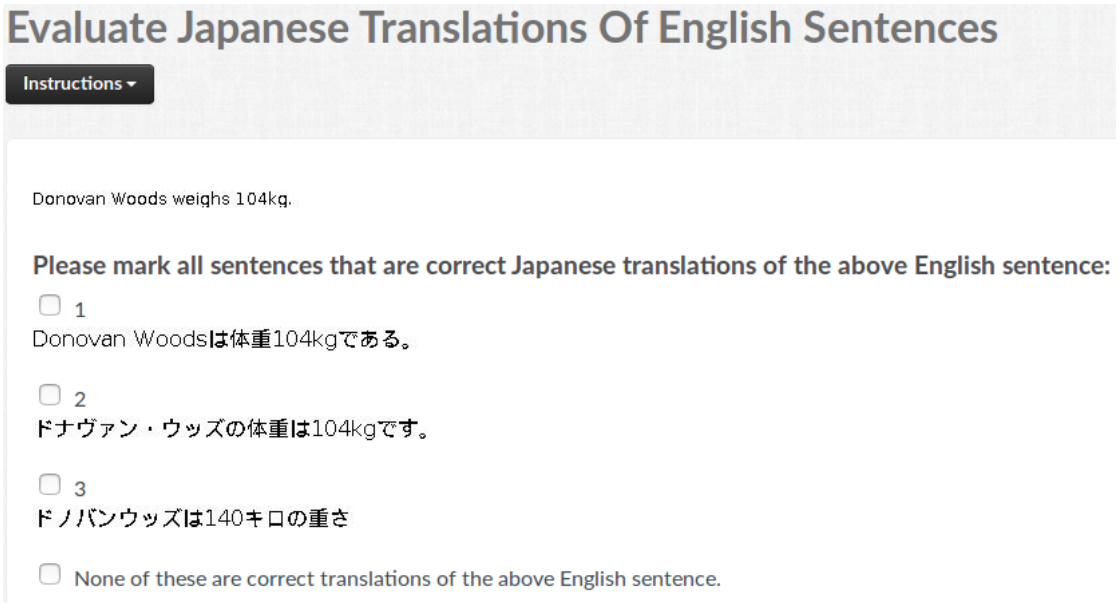


Figure 11: Example task from the evaluation stage of our test run

We generated our test questions from English seed sentences we had already used in the translation stage. Each such sentence we combined with clearly correct translations from the preceding translation stage and/or randomly chosen translations of other sentences. Furthermore, for some test questions we added manual translations of our own in which we had included grammatical errors.
In total, we did two separate runs of the evaluation stage, one with Japanese instructions and one with English instructions, to test again if the language of instructions has any effect on the received results. Some basic data for both runs are shown in Table 4. To check if the test questions are efficient at filtering out workers with low-quality contributions, we looked at a number of normal microtasks for which at least for some of the translations it was clear whether they are correct or not, and looked at how the workers who passed the test questions performed on these. In many respects, the two runs proceeded very similarly: Both took considerably shorter than the runs of the preceding translation stage, and both were also considerably cheaper. Also in both cases, a large number of people attempted to work on the task, but failed at the test questions. Workers who feel that the test questions are incorrect or unfair can give feedback to the employer. In three cases we received feedback that lead us to deactivate the respective test questions, as at closer inspection of these we had the impression

	First run	Second run
Language of instructions	Japanese	English
Total number of workers	46	29
... who passed test questions	10	8
Low-quality contributors	2	3
Duration	7 days	3 days
Maximum cent/sentence	1	1

Table 4: Basic data of the first and second run of the evaluation stage

that the criticism was justified. However, the vast amount of workers who did not pass the test questions did not give any feedback at all, and the performance of many of these looked as if they had simply answered randomly. Hence, one insight from these test runs seems to be that a lot of people will try to work on tasks they are clearly not competent for. Furthermore, it looks as if the test questions are efficient at filtering out people who do not provide acceptable results. Given the small overall amount of people who actually passed the test questions, it is difficult to make any definite statements about the differences between the two runs with respect to this group. For the second run the amount of people who passed the test is slightly lower, while the number of people out of this group who still delivered low-quality contributions is slightly higher. However, with only three days this run also took less time than the first one.

4.4 EVALUATION

4.4.1 *Evaluation Task vs. Majority Decision*

Prior to our crowdsourcing experiments we had manually compiled a Japanese gold standard lexicon for the ten exemplary ontology elements from table 2 we had used in our crowdsourcing test run. Tables 5 and 6 show the results of comparing this gold standard against a number of different subsets of the candidate verbalizations retrieved during the translation stage of the test run. These subsets were formed according to

1. the number of translations a given candidate verbalization occurred in (table 6), and
2. the number of upvotes a verbalization received during one of the runs of the evaluation stage (table 5).

This way one can see if and how the evaluation stage of our workflow can actually improve the quality of the resulting lexicon, or if a simple majority decision based on the number of translations a candidate verbalization occurs in would be sufficient to generate lexicons of good quality. As can be seen from the tables, for the whole crowdsourced lexicon, whose results are shown in the uppermost row respectively, precision is only at around 71 percent, and by means of either validation mechanism, precision can be raised to nearly hundred percent. This shows that the results from the translation stage should be postprocessed or filtered in some way. However, since both evaluation strategies yield nearly the same results in terms of precision, and the evaluation based on majority decision does not require a further crowdsourcing stage, but only information from the translation stage itself, at least for the specific settings of our test run the evaluation stage of our workflow could have been left out. Out of the 49 verbalizations given in the gold standard, eleven do not occur in the result set of our test run at all; therefore, even for the whole set of candidate verbalizations from the translation stage recall is only at 0.77. One possible reason may be the influence the English seed sentences have on the syntactic constructions — and therefore parts-of-speech — and semantic level of granularity workers will use in their translations. For example, for the property `yearOfConstruction` the Japanese gold standard, among other entries, contains the construction 完成する（に） (*kanseisuru* (*-ni*)), which is a rather general term that could be translated as 'to complete (in)'. However, the English gold standard we worked with contained the more specific 'constructed (in)' as the

only verbalization of `yearOfConstruction`. Accordingly, workers were only shown sentences of the form *X was constructed in [year] Y* for this ontology element, and we only retrieved Japanese verbalizations at around the same level of semantic specificity. In most cases, filtering based on the votes from the first run of the evaluation stage yields better recall and better F-scores, though not always better precision, than filtering based on the votes from the second run. However, overall the quality of the votes from both runs seems to be very similar, which would back our finding from the translation stage that the language the task instructions are given in does not have much effect on the quality of the retrieved results.

4.4.2 *Alternative Workflow*

Since it seems that the evaluation stage could be replaced with a simple majority decision, as an approach to further crowdsourcing experiments it may be worthwhile to try out an alternative workflow as shown in figure 12, in which the results of the translation stage get validated based on the number of translations they occur in, instead of based on the results of an evaluation stage. Sentences that contain candidate verbalizations which have been validated this way as correct could then be used as input to a further crowdsourcing task in which workers are asked to provide paraphrases to these sentences. This paraphrasing task may help in retrieving a wider variety of different verbalizations for one given ontology element, and may therefore help to achieve a better recall of the crowdsourced lexicon. However, there are still a number of open questions with regards to this alternative workflow, such as how the results from the paraphrasing task should be quality-controlled.

4.4.3 *Costs*

The English seed lexicon contains 1,217 entries in total, which — given the settings of our test run — would amount to 10,953 microtasks for both the translation and evaluation stage. Hence, at a payment of one cent per microtask, carrying out the evaluation stage for the whole lexicon would cost 109.53 dollars. The translation stage would cost something between that same amount and 876.24 dollars in case we have to pay eight cent for every translation.

# Votes	Precision	Recall	F-score
≥ 0 (=all)	0.71	0.77	0.74
Votes from first run of evaluation stage (Japanese instructions)			
≥ 1	0.72	0.77	0.74
≥ 2	0.83	0.74	0.78
≥ 3	0.88	0.71	0.79
≥ 4	0.99	0.64	0.78
Votes from second run of evaluation stage (English instructions)			
≥ 1	0.79	0.77	0.78
≥ 2	0.82	0.69	0.75
≥ 3	0.88	0.68	0.77
≥ 4	0.99	0.61	0.75

Table 5: Precision, recall and F-score of different subsets of the crowdsourced lexicon formed according to the number of votes each candidate verbalization received during the evaluation stage.

# Translations	Precision	Recall	F-score
≥ 1 (=all)	0.71	0.77	0.74
≥ 2	0.85	0.69	0.76
≥ 3	1.0	0.60	0.75

Table 6: Precision, recall and F-score of different subsets of the crowdsourced lexicon formed according to the number of translations each candidate verbalization occurred in.

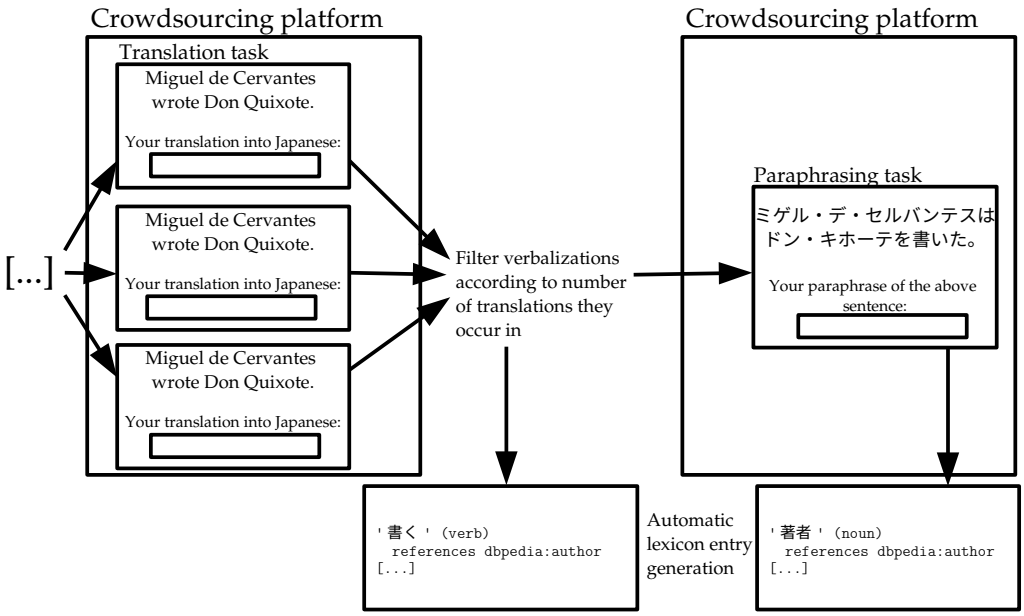


Figure 12: Alternative workflow for further crowdsourcing experiments

4.4.4 *Outlook*

The scale of this first test run is sufficient to show that crowdsourcing ontology lexicons in general and our workflow presented above in particular are basically feasible. Still, a number of questions that occurred during our work, e.g. concerning the appropriate amount of payment during the translation stage or the effect the language of instruction has, would require more data to answer with confidence. Therefore, further tests at a larger scale — e.g. based on a larger number of exemplary ontology elements — would be due. However, given the current changes on CrowdFlower described in Section 4.3.3, we would most probably have to look for a new crowdsourcing platform: While the evaluation stage of our current test run was quite successful, both the outcome of our first run of the translation stage and the large amount of people who attempted to work on the evaluation stage, but failed at the test questions, show that attempting to re-run the translation stage on CrowdFlower would not make much sense as long as we can no longer narrow down the set of allowed workers. One possible alternative would be to switch to a Japanese crowdsourcing platform such as Yahoo! Crowdsourcing[8].
In summary, the outcome of the test run seems to suggest that crowdsourcing is a viable option for generating ontology lexicons, given that an English seed lexicon is already available. However, further experiments would be necessary to determine which workflow would be optimal.

[8] `http://crowdsourcing.yahoo.co.jp/`

5

ADAPTING M-ATOLL TO JAPANESE

5.1 OVERVIEW

As mentioned before, M-ATOLL comprises three separate approaches to the generation of ontology lexicon entries: The label-based approach, which can be used both for classes and properties, the corpus-based approach, which works for properties only, and an approach to the generation of adjective entries based on machine learning. The main challenge when adapting the label-based approach, which generates verbalizations based on the ontology-internal labels of the given class or property, is the lack of Japanese labels in DBpedia; in section 5.3.2 we will talk about different approaches to overcoming this label sparseness. We dealt with the corpus-based approach first, since it seems to be the most language-dependent one out of these three. Figure 13 shows the overall architecture of this approach. It comprises the following steps:

1. Before the sentences from the corpus at hand can be passed to M-ATOLL, they need to be dependency-parsed. Depending on the language under consideration and the dependency parsers available for it, different types of information may be given in the dependency parses. One has to decide which of these types of information are relevant for M-ATOLL and should hence be present in the input to the system.

2. The sentence preprocessing component of M-ATOLL selects all sentences for further processing that contain labels of entities that are linked in the given knowledge base by the property one wishes to retrieve verbalizations for, and marks those nodes in the dependency parse that correspond to these labels. One has to decide here which entity labels one wants to work with: As was shown in table 1 for DBpedia, depending on the language at hand only very few language-specific entity labels may be available, so one may decide to use further labels which were either specified for other languages or which come from further, external resources. Furthermore, there may be language-specific aspects to the identification of the nodes that correspond to the entity labels. For example, in Japanese numbers can be given either as Arabic or Chinese numerals — or as a

mixture of both — and hence mapping the objects of datatype properties onto the corresponding elements in Japanese sentences may be more complicated than e.g. for English.

3. Each sentence selected for further processing in the preceding step is then matched against a number of language-specific SPARQL queries, each of which corresponds to a dependency relation regularly holding between property verbalizations and entity labels. An example for English, the pattern `Transitive_Verb` is given in figure 14. Obviously, adapting this step to another language requires generating a set of SPARQL queries for the new language at hand.

4. Finally, if a (part of a) sentence is a match for one of the SPARQL queries, a lexicon entry is generated for it. M-ATOLL allows for different kinds of entries, depending on the SPARQL query the sentence was matched by and hence the (sub-)part-of-speech of the candidate verbalization at hand. These different kinds of entries are defined in terms of so-called templates; for example, the English `Transitive_Verb` query invokes the template `TransitiveVerb`, an example entry for which is shown in figure 14. Here, for a new language to be used with M-ATOLL one needs to decide which of the existing templates may be suitable for presenting information about lemmas in the new language, or if one needs to define new templates for the language at hand.

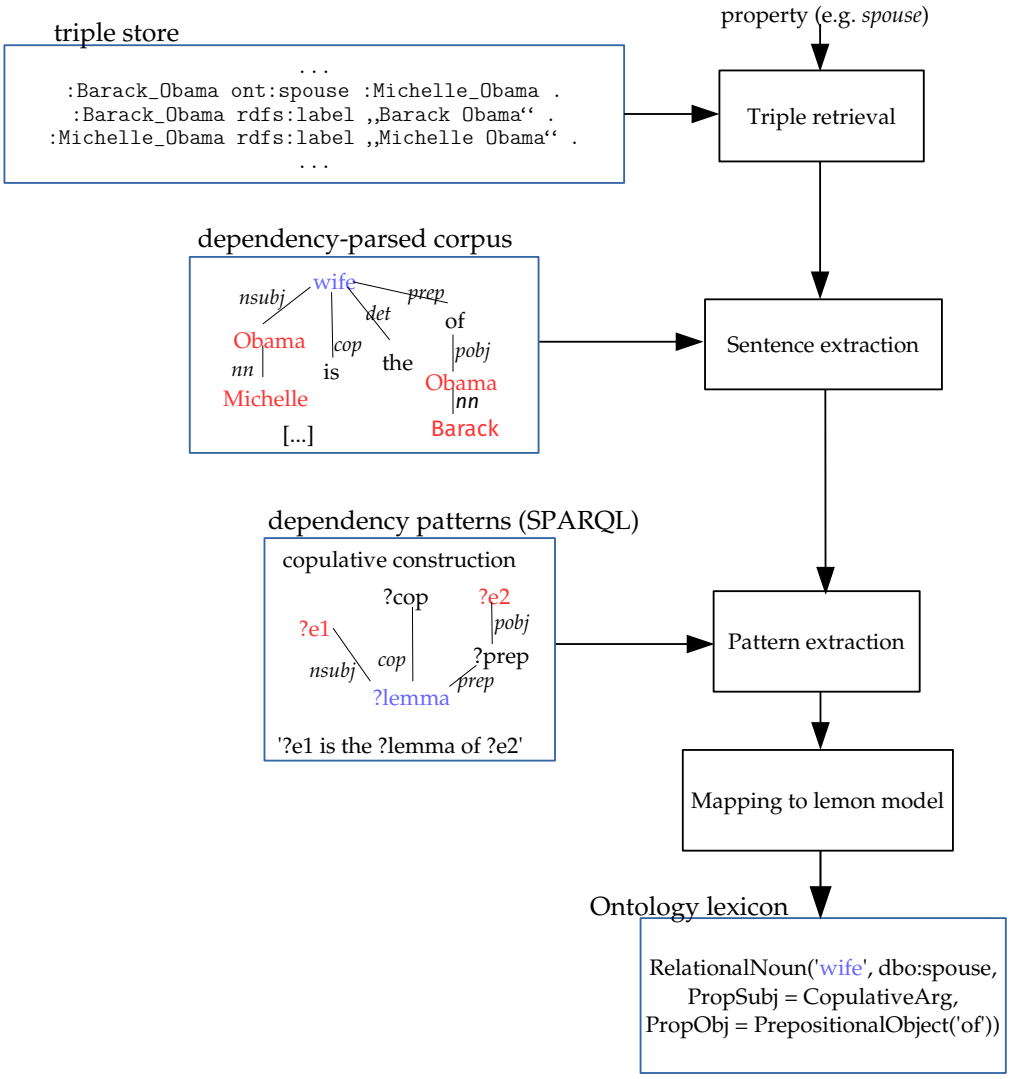
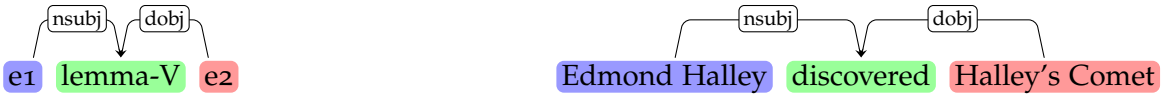


Figure 13: M-ATOLL's corpus-based approach to the generation of verbalizations



- canonical form: *discover*
- part-of-speech: verb
- subcategorization frame: transitive frame
  arguments:
  - subject e_1
  - direct object e_2
- semantic reference: `discoverer`
  arguments:
  - subject e_1
  - object e_2

Figure 14: SPARQL query `EN_Transitive_Verb` that matches English transitive verbs in active voice (Walter 2017, p. 131), plus lexicon entry for the example on the top right generated through template `TransitiveVerb`

.

5.2 INPUT FORMAT

Since we want to port the corpus-based approach to the Japanese Wikipedia, the source data for generating the input for M-ATOLL are the texts from the Japanese Wikipedia in XML format, which can be downloaded from the site of the Wikimedia Foundation[1]. We extract the sentences from the XML file with an already existing script[2]. Next, we replace each token delimiter in the sentences with a hash (#). This concerns on the one hand blank characters, which may occur e.g. in foreign proper nouns given in rōmaji (*Donald Trump*), and nakaguros (・), which among other uses serve to indicate token boundaries in proper nouns given in katakana (ドナルド・トラムプ). The different types of delimiters are not treated uniformly by the different tools required by M-ATOLL; for example, the morphological analyzer we use removes blank characters, but keeps nakaguros. Furthermore, there exist two different variants of the nakaguro — a full-width and a half-width version — which some tools treat as identical, while for other tools these are different characters. Hence, in order to allow for a uniform treatment of all token delimiters and an overall simpler workflow, we decided to replace all token delimiters. We then run the morphological analyzer MeCab[3] on the modified sentences, which splits them into their single tokens and provides further information about the tokens such as their part-of-speech. The result of MeCab is again used as the input to the dependency parser J.DepP[4]; its output for the following example is shown in figure 15.

(18) *1943*年、 ロスアラモス国立研究所-を
senkyuuhyakkuyonjuusannen *rosuaramosukokuritsukenkyuujo-o*
1943.year Los.Alamos.National.Laboratory-DOBJ
建設した。
kensetsushita
constructed
In 1943, [someone] constructed the Los Alamos National Laboratory.

As was already mentioned in section 2.7, in contrast to most parsers for Indo-European languages Japanese dependency parsers generate dependency structures that do not hold between single tokens, but between multi-word units called *bunsetsus*. For example, in figure 15 multi-word unit 0, which contains the tokens 1943, 年 (*nen*) and a comma, depends upon multi-word unit 2, which consists of the tokens 建設 (*kensetsu*), し (*shi*),

[1] https://dumps.wikimedia.org/jawiki/
[2] http://medialab.di.unipi.it/wiki/Wikipedia_Extractor
[3] http://taku910.github.io/mecab/
[4] http://www.tkl.iis.u-tokyo.ac.jp/~ynaga/jdepp/

```
# S-ID: 656657; J.DepP
* 0 2D
1943     名詞,数,*,*,*,*,*
年           名詞,接尾,助数詞,*,*,*,年,ネン,ネン
、           記号,読点,*,*,*,*,、,、,、
* 1 2D
ロスアラモス          名詞,一般,*,*,*,*,*
国立           名詞,一般,*,*,*,*,国立,コクリツ,コクリツ
研究所           名詞,一般,*,*,*,*,研究所,ケンキュウジョ,ケンキュージョ
を           助詞,格助詞,一般,*,*,*,を,ヲ,ヲ
* 2 -1D
建設           名詞,サ変接続,*,*,*,*,建設,ケンセツ,ケンセツ
し           動詞,自立,*,*,サ変・スル,連用形,する,シ,シ
た           助動詞,*,*,*,特殊・タ,基本形,た,タ,タ
。           記号,句点,*,*,*,*,。,。,。
EOS
```

Figure 15: Output of dependency parser J.DepP for example sentence 18

た (*ta*) and a full stop. The grammatical information provided for each token comprises up to four part-of-speech tags of differing granularity, the inflection class and the given inflection form in case of verbs and adjectives, the base form of the token, its reading, and its pronunciation. Table 7 shows those part-of-speech and inflection type tags that were used in the formulation of the SPARQL queries later on. In the next step, we remove all punctuation marks from the parsed sentences, which facilitated writing the SPARQL queries in a subsequent step.

One of the tasks of M-ATOLL's sentence preprocessing component is to turn the input sentences into RDF. Since the Malt parser[5], which had been used for dependency parsing the English and Spanish input to M-ATOLL (Walter 2017, p. 144), uses the CoNLL format[6] as its output format, the sentence preprocessing component was already able to deal with this format and turn it into RDF. Furthermore, a token-based dependency structure allows one to use both dependency relations among bunsetsus and among tokens in the specification of one's SPARQL queries, and in order to keep both options available, we wanted to transform the bunsetsu-based dependency

[5] http://www.maltparser.org/userguide.html
[6] http://ilk.uvt.nl/conll/

part-of-speech	part-of-speech subcategory 1	inflection type
名詞 (*meishi*) 'noun'	サ変接続 (*sahensetsuzoku*) 'verbal' (nouns that can form verbs by being followed by する (*suru*) or related verbs)	
動詞 (*doushi*) 'verb'	自立 (*jiritsu*) 'main' (i.e. non-auxiliary)	特殊・デス (*tokushu desu*) 'copula verb です (*desu*)' 特殊・ダ (*tokushu da*) 'copula verb だ (*da*)'
助詞 (*joshi*) 'particle'	係助詞 (*kakarijoshi*) 'dependency' (comprises topic marker は (*wa*)) 連体化 (*rentaika*) 'adnominalizer' (non-possessive の (*no*) that joins nouns together)	

Table 7: Part-of-speech and inflection type tags used in the SPARQL queries.

structure into a token-based one, which could be better represented in the CoNLL format. Hence, we decided to transform the original output format of J.DepP into a modified version of the CoNLL format for further processing.
The dependency parse of example sentence 18 is again shown in figure 16, this time in the CoNLL format. Table 8 shows a comparison between the features occurring in J.DepP's output and those employed in the CoNLL format. One of the main differences between the two formats is that in the CoNLL format instead of multi-word units each single token is assigned an index, and dependency relations hold between tokens. When transferring the J.DepP format into the CoNLL format we had to generate the token indices (`ID`) from the tokenization provided by MeCab. In contrast, `FORM` and `LEMMA` could be directly mapped from the respective columns in the J.DepP format. For the `CPOSTAG` and `POSTAG` columns we used the main part-of-speech tag column from the J.DepP format (column 2) and the first sub-part-of-speech tag column (3), respectively; hence, the information about the other two part-of-speech subtypes was lost in the transformation, which we considered not that problematic since most of the time those two columns are empty anyway. The information about inflection classes and forms (columns 6 and 7) was merged into the `FEATS` column in the CoNLL format separated by

a vertical bar. In order to generate the correct values for the HEAD column, the following rules were used in order to transform the original bunsetsu-based dependency structure into a token-based one:

- If in the original dependency-based structure bunsetsu b_1 depends upon bunsetsu b_2, then in the token-based structure the last token of b_1 depends upon the last token of b_2.
- If a token belongs to a bunsetsu b and is not the last token within that bunsetsu, it depends upon the token that directly follows it within b.

As an example, figure 17 shows how the bunsetsu-based dependency structure of sentence 18 would be transformed into a token-based structure. The remaining columns of the CoNLL format were left empty: As mentioned before in section 2.7, most Japanese dependency parsers such as J.DepP do not assign labels to the dependencies, hence no information was available for the DEPREL column. The remaining two columns seem to serve no real purpose in the context of M-ATOLL; at least they are referenced nowhere in the SPARQL queries for the Indo-European languages. The last two columns of the J.DepP format, which contain information about the reading and the pronunciation of the token at hand, were discarded in the transformation process, as this kind of information did not seem very relevant to the purpose of an ontology lexicon.

In order to keep the information about which tokens belong to which bunsetsu, we adopted the representation of multi-word units used in the CoNLL-U format[7], which is a revised version of CoNLL aimed at being able to represent a larger variety of different languages[8]: Multi-word units are given in addition to the tokens they are comprised of, and instead of a single index they are assigned a range of indices, as shown in figure 16. The remaining features are not specified for multi-word units.

[7] http://universaldependencies.org/format.html
[8] http://universaldependencies.org/introduction.html

J.DepP		CoNLL	
field number	field name/descr.	field number	field name/descr.
1	surface form	1	ID (token counter, starting at 1 for each new sentence)
2	part-of-speech	2	FORM (word form/punctuation symbol)
3	part-of-speech, subtype 1	3	LEMMA (lemma or stem; underscore if not available)
4	part-of-speech, subtype 2	4	CPOSTAG (coarse-grained part-of-speech tag)
5	part-of-speech, subtype 3	5	POSTAG (fine-grained part-of-speech tag)
6	inflection class (for verbs and adjectives)	6	FEATS (set of morphological and/or syntactic features, separated by |, underscore if not available)
7	inflection form (for verbs and adjectives)	7	HEAD (head of the current token; either a value of ID or zero)
8	lemma	8	DEPREL (type of the dependency relation to the head)
9	reading	9	PHEAD (projective head of the current token; either a value of ID, zero, or underscore if not available)
10	pronunciation	10	PDEPREL (type of the dependency relation to the projective head; underscore if not available)

Table 8: Types of information present for each token in the output format of MeCab/J.DepP (`http://taku910.github.io/mecab/`) and in the CoNLL format (Walter 2017, 29)

```
1     1943          _          名詞          数          _|_     2          _
2     年     年          名詞          接尾          _|_     9          _
3     ロスアラモス     _     名詞     一般     _|_     4     _
4     国     立     国     立          名     詞          一     般
_|_     5          _
5     研     究     所     研     究     所          名     詞          一     般
_|_     6          _
6     を     を          助詞          格助詞          _|_     9          _
7     建     設     建     設          名     詞          サ     変     接     続
_|_     8          _
8     し     する     動詞     自立     サ変・スル|連用形     9     _
9     た     た     助動詞     _     特殊・タ|基本形     0     _
1-3     米国政府は _          _          _          _          _          _
4-5     1943年 _          _          _          _          _          _
6-11     第二次世界大戦の          _          _          _          _          _          _
12-13     最中に          _          _          _          _          _          _
14-17     ロ     ス     ア     ラ     モ     ス     国     立     研     究     所     を
_          _          _          _          _          _
18-20     建設した          _          _          _          _          _          _
```

Figure 16: Output of dependency parser J.DepP for example sentence 18, turned into CoNLL format

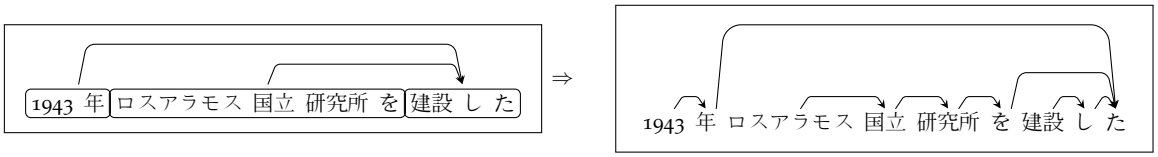


Figure 17: Exemplary transformation of bunsetsu-based into token-based dependency structure. The boxes indicate bunsetsu boundaries.

5.3 SENTENCE EXTRACTION AND PREPROCESSING

5.3.1 *Introduction*

As a starting point, we first took the Japanese sentences we had received through the crowdsourced translation task and tried to adapt M-ATOLL's sentence preprocessing component so that as many of those sentences as possible were selected for further processing. That is, ideally, the sentence preprocessing component should recognize all sentences that contain labels of entities that are linked in DBpedia's triple store by one of the properties we had used for the crowdsourcing task.

5.3.2 *Choice of Entity Labels*

As has been shown in table 1, for languages other than English only few elements in DBpedia's ontology have an according label. Hence, when M-ATOLL is to be used with one of these languages and only the respective language's labels are used in the sentence preprocessing step, only very few sentences that contain references to entities that are linked by the property at hand may be found. Accordingly, in the beginning many of the sentences we had received through crowdsourcing were not selected for further processing by the sentence preprocessing component simply because for either one or both of the entities mentioned in the sentence at hand there was no Japanese label available that could have matched. As a first step towards alleviating this problem, apart from the Japanese labels also the English ones are used for preprocessing Japanese: For a given entity pair the sentences are searched for any combination of Japanese and/or English labels available for the two entities. For languages with a Latin-based writing system such a solution may already be sufficient, since at least proper nouns will most probably be written in the same way as in English. For example, in the English sentence 19 and the German sentence 20 the two entities Fred Trump and Donald Trump are denoted in the same way, and one could use identical sets of labels to match those entities in both sentences. In Japanese, however, proper names are most of the time not given in rōmaji but e.g. in case of foreign names in katakana, as shown in sentence 21. Therefore, in many cases English labels will not be useful for detecting entities in Japanese texts.

(19) Fred Trump is the father of Donald Trump.

(20) Fred Trump ist der Vater von Donald Trump.

(21) フレッド・トランプはドナルド・トランプの父である。
(*fureddo toranpu wa donarudo toranpu no chichi dearu*)

One possible way to retrieve further Japanese entity labels is to search the Japanese Wikipedia for anchor texts and use these as additional labels for the entity that corresponds to the article the respective anchor text links to. Since this approach can only provide labels for entities for which a corresponding article exists in the Japanese Wikipedia, and since these entities usually already have a Japanese label in DBpedia, this approach may be helpful for finding further labels in addition to the already existing ones. However, it does not help in the majority of cases where no Japanese label exists at all. In case of our crowdsourced sentences, this approach did not help us to match any further entities. Another approach would be to automatically transfer the English label into Japanese. We attempted to automatically transfer the English labels into katakana by means of a string transformation; however, the results were of very poor quality and again no further entities could be matched in the crowdsourced sentences: Katakana is actually a form of transliteration, i.e. it takes the pronunciation in the source language into account, while a simple string transformation can only produce a transcription, i.e. a transformation based solely on written characters. While there are a number of web sites that offer a pronunciation-based conversion from English to katakana[9], these sites do not allow programmatic access, and for larger amounts of English labels typing them separately into a web formular would not be feasible. In the end, we used Google Translate to generate Japanese labels for those entities used in crowdsourced sentences that did not have a Japanese label yet. These translated labels turned out to be of good quality and helped in matching most sentences with entities that did not have a Japanese label before. Furthermore, due to the availability of the Google Translate API[10] in principle this solution could also be used to translate larger amounts of English labels.

5.3.3 *Datatype Properties*

As table 9 shows, M-ATOLL's sentence preprocessing component selects considerably less sentences for further processing in case of datatype properties than in case of object properties. In particular the two datatype properties `numberOfStudents` and `weight` stand out in that only three or even only one sentence, respectively, are recognized

[9] e.g. `http://www.sljfaq.org/cgi/e2k.cgi`
[10] `https://cloud.google.com/translate/`

property	#triples	#sentences	ratio #sent.s/#triples	datatype/object property?
`parent`	28,295	11,831	0.4181	o
`occupation`	330,322	9,531	0.0289	o
`colourName`	7,768	120	0.0154	d (rdf:langString)
`budget`	17,093	200	0.0117	d (xs:double)
`yearOfConstruction`	46,702	281	0.0060	d (xsd:gYear)
`numberOfStudents`	28,233	3	0.0001	d (xsd:nonNegativeInteger)
`weight`	74,059	1	<0.0001	d (xsd:double)

Table 9: The number of triples available in DBpedia's triple store plus the number of sentences matched by M-ATOLL's sentence preprocessing component for each DBpedia property that was used in the crowdsourcing task. For datatype properties also the type of their range, i.e. the type of objects they take, is given.

as containing references to an entity pair that is linked by the respective property in DBpedia's triple store.

A number of different factors seem to contribute to this behavior: First of all, in case of datatype properties that take numeric objects often the units of measurement used within DBpedia differ from those occurring in natural language sentences. For example, in case of the `weight` property the triples contain the weight of a given entity in gram, while e.g. the weight of humans is usually given in kilograms or in pounds in texts. In order to solve this problem one would need to transform the values given either in the triple store or in the corpus into the unit of measurement used in the other resource, at worst for many different numeric datatype properties separately, which seems extremely laborious. A further reason that may apply specifically to Wikipedia is that here, information covered by datatype properties seems to be often given in form of an infobox that says e.g. 'weight: 56kg' instead of a full sentence like 'Person X weighs 56kg'. Finally, a problem specific to Japanese is that in Japanese texts numbers may be given in Chinese or Arabic numerals, or as a mixture of both. For example, the following sentences all contain acceptable writing variants of the numeral 1,365,000:

(22) 一百三十六万五千*USD*-は
hyakkusanjuurokumangosendoru-wa
one.hundred.three.ten.six.tenthousand.five.thousand.USD-TOP
ニノチカ-の 予算 である。
ninochika-no *yosan* *dearu*
Ninotschka-POSS budget COP
1,365,000USD is the budget of Ninotschka

(23) 136万5000USDはニノチカの予算である。

(24) 1365000USDはニノチカの予算である。

Therefore, when searching for references to numeric objects of datatype properties in the given corpus, one would need to be able to turn (partly) Chinese numerals into Arabic ones, which could in principle be done through Lucene's[11] `JapaneseNumberFilter`. Normalizing the numerals after dependency parsing would not be feasible, since different parts of a given Chinese numeral may end up in different nodes of the dependency path, while Arabic numerals are stored in one node only; hence, modifying the numerals after dependency parsing would require modifying the dependency tree. However, running the Japanese number filter before dependency parsing may also lead to problems, since for some reason it removes all punctuation marks from sentences, which are necessary for dependency parsing.

For now, we have concentrated on adapting M-ATOLL to the generation of lexicon entries for object properties and certain datatype properties to which the problems just mentioned do not apply.

[11] http://lucene.apache.org/core/

5.4 SPARQL QUERIES FOR JAPANESE

5.4.1 *Pattern Generation Process*

We defined eleven dependency patterns for Japanese in terms of SPARQL queries, which are presented in tables 10 and 11. Six of these patterns serve to retrieve noun lemmas, while the remaining five match verbs. We have not yet dealt with adjective lemmas and the respective SPARQL queries; however, some of the patterns for nouns should be able to match noun-like adjectives, as described in section 2.6.

As a starting point, we first created SPARQL patterns based on the sentences we retrieved through crowdsourcing for the object properties `parent`, `occupation` and `colourName` and the datatype property `yearOfConstruction`. The latter property takes a year date as its object; the problems with many other datatype properties described in the preceding section, such as differences in the systems of measurement used in the triple store and the corpus at hand, plus the issues with Japanese numerals, do not apply here. Furthermore, the dependency patterns in which this property gets verbalized in the crowdsourced sentences are very similar to those in which verbalizations of object properties occur. Hence, `yearOfConstruction` was considered even though it is a datatype property.

The English seed sentences we had used in the crowdsourced translation task all either had the copula structure *[e1] is the [lemma] of [e2]* in case of noun lemmas ('Lydia Hearst is the child of Patty Hearst') or the structure *[e1] [lemma]s ([prep]) [e2]* in case the given verbalization from the English seed lexicon was a verb ('Chris Douglas works as songwriter'). The crowdsourced translations stuck to the Japanese equivalents of these structures for the most part, which gave us the SPARQL patterns `MainCopulaCompl` and `MainVerb`. Additionally, sometimes seed sentences with a noun verbalization were translated into a slightly different copula structure that in English would correspond to *The [lemma] of [e2] is [e1]* (e.g. ボブ・ショーの職業はジャーナリスト (*bobu shoo no shokugyou wa jaanarisuto*) 'The occupation of Bob Shaw [is] journalist'), which resulted in the `MainCopulaSubj` pattern. Since the sentences these patterns are based on were retrieved through a translation task, they may not necessarily represent the most frequent patterns for Japanese; however, they seemed prototypical enough to be included in our final set of SPARQL patterns.

The remaining patterns were retrieved based on three of the properties from the crowdsourcing task — `parent`, `occupation` and `yearOfConstruction` — and two new

properties, `crosses` and `nationality`. The approach to generating the remaining patterns was similar to that described in Walter (2017, p. 55):

1. For a given property, we extracted all sentences from the Japanese Wikipedia that contain labels of entity pairs which are linked by the respective property in DBpedia's triple set.

2. Furthermore, we generated a set of gold verbalizations our SPARQL patterns should be able to find. For the properties we had already used in the crowdsourcing task we simply used the verbalizations we had retrieved this way.

3. We then searched the sentences from 1) for occurrences of these gold verbalizations. We looked at the dependency constructions they were embedded in, and watched out for frequently occurring patterns.

For example, we first looked at all sentences that contain on the one hand labels of entities that are linked by the property `parent` in DBpedia's triple store and on the other hand one of the verbalizations we had received through crowdsourcing for that property. This way, we found a number of sentences in which the entity label pairs and the verbalizations occur in the same kind of construction, such as the following:

(25)

ヘンリー2世-の	母親	である	皇后	マティルダ-は	これ-に
henriinisei-no	*hahaoya*	*dearu*	*kougou*	*matiruda-wa*	*kore-ni*
Henry.II-POSS	mother	COP	empress	Mathilda-TOP	this-IOBJ

反対した
hantaishita
opposed

Empress Mathilda, who is the mother of Henry II, opposed this

(26)

宮崎吾朗-の	父親	である	宮崎駿-は
miyazakigorou-no	*chichioya*	*dearu*	*miyazakihayao-wa*
Goro.Miyazaki-POSS	father	COP	Hayao.Miyazaki-TOP

『ゲド戦記』-の	古く-から-の	ファン	である
gedosenki-no	*furuku-kara-no*	*fan*	*dearu*
"Earthsea"-POSS	ancient-from-POSS	fan	COP

Hayao Miyazaki, who is the father of Goro Miyazaki, is an old fan of "Earthsea"

This structure also reoccurred for other properties, such as for `crosses` in the following example, which gave us confidence that it is indeed a general, not property-specific construction that should be incorporated in the set of dependency patterns M-ATOLL

uses for Japanese. In general, when a given structure could only be found in sentences for one particular property, we decided based on intuition whether it may be a general or a property-specific structure.

(27) 木曽川-の 橋 である 愛岐大橋-は 慢性的な
kisogawa-no *hashi* *dearu* *aigioohashi-wa* *manseitekina*
Kiso.river-POSS bridge COP Aichi.Bridge-TOP frequent
渋滞-が 発生している
juutai *hasseishiteiru*
congestion-SUBJ was.happening
on Aichi Bridge, which is a bridge of the Kiso river, frequent congestions were happening

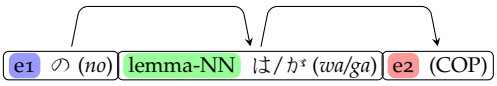
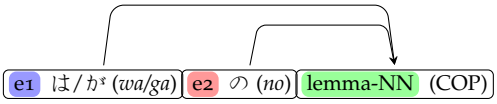
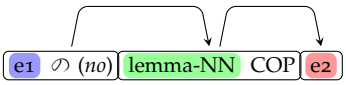
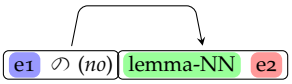
As a result, the respective pattern was added to the set of SPARQL queries as `RelCopulaCompl`.
Some of the SPARQL queries we generated this way could rather easily be merged with others; for example, `MainVerbPtclEllipsis` is nearly the same as `MainVerb`, only that in the former one particle is left out. We decided against merging patterns in cases were we were not sure whether one of the given patterns was really productive enough to be kept in the set of SPARQL queries; for example, the pattern `MainVerbPtclEllipsis` was only encountered for the property `yearOfConstruction`, and we were not sure how productive particle ellipsis actually is. Keeping such patterns as separate queries makes it easier to check during the analyses carried out in section 5.6 whether they contribute positively to the lexicon or not; if so, they may still be merged with another pattern, but if not, it will be easier to remove them this way.
The manual approach to the generation of SPARQL queries just described is rather laborious and time-consuming. In subsection 5.6.1 we will present a first step towards automatizing the search for SPARQL queries.

5.4.2 *Noun Patterns*

Similarly to what has been described for English, German and Spanish in Walter (2017, p. 55), most patterns we were able to identify for noun lemmas correspond either to an appositive or a copula construction. While in Japanese a number of different constructions may be considered appositions (Heringa 2012), we only came along one of these construction types, where anchor and apposition are placed directly alongside. We generated two different patterns (`Apposition` and `MainCopulaAppos`) in which the apposition is embedded into one of its two most commonly occurring syntactic contexts,

respectively, since the single, very general apposition pattern we used at first produced a lot of noise. Apart from the copula constructions already discussed in the preceding section and the two appositive constructions, we only found one further pattern that did not belong to either of these two groups (`RelDobj`), in which the lemma occurs as a direct object of a relative clause that contains the first entity as a further participant and has the second entity as its head.

Both here and in case of the verb patterns, relative clause patterns such as `RelCopulaCompl` only match if the relative clause at hand does not contain an overt subject. In accordance with the heuristic described in section 2.6, it is then assumed that the head of the relative clause — i.e. the label of the second entity — acts as the clause's subject.

Name	Dependency structure	Example	Can be merged with
`MainCopulaSubj`		ボブ・ショー-の *bobushoo-no* Bob.Shaw-POSS 職業-は *shokugyou-wa* occupation-TOP ジャーナリスト *jaanarisuto* journalist The occupation of Bob Shaw is journalist.	-
`MainCopulaCompl`		バーブラ・ストライサンド-は *baaburasutoraisando-wa* Barbara.Streisand-TOP ジェイソン・ゴールド-の *jeisonguurudo-no* Jason.Gould-POSS 母 である *haha dearu* mother COP Barbara Streisand is the mother of Jason Gould.	-
`RelCopulaCompl`		ヘンリー2世-の 母親 である *henriinisei-no hahaoya dearu* Henry.II-POSS mother COP 皇后 マティルダ *kougou matiruda* empress Mathilda Empress Mathilda, who is the mother of Henry II	-
`Apposition`		エンリケ-の 息子 *enrike-no musuko* Enrique-POSS son アフォンソ1世 *afonsoissei* Afonso.I Enrique's son Afonso I	-

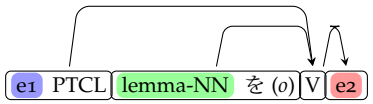
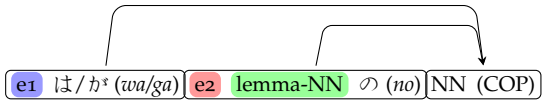
RelDobj		1764年-から *sennanahyakkurokujuuyonen-kara* 1764.year-from 建設-を 開始した *kensetsu-o* *kaishishita* construction-DOBJ started モントピリア *montopiria* Montpelier Montpelier, whose construction started in 1764	RelVerb
MainCopulaAppos		ルイ・リュシアン・ボナパルト-は *ruiryushianbonaparuto-wa* Louis.Lucien.Bonaparte-TOP フランス 国籍-の *furansukokuseki-no* France.nationality-POSS 言語学者 *gengogakusha* linguist Louis-Lucien Bonaparte is a linguist of French nationality	-

Table 10: SPARQL queries for noun lemmas. The boxes indicate bunsetsu boundaries.

5.4.3 *Verb Patterns*

For English, German and Spanish, there are separate patterns for transitive and intransitive verbs, as well as for verb occurrences in the active and passive voice, respectively (Walter 2017, 131-136). For Japanese, in contrast, due to the use of particles to mark all grammatical functions alike, and the way the passive voice gets marked simply through an auxiliary, one does not necessarily need to differentiate between patterns for transitive and intransitive verbs, and patterns for verbs in the active and passive voice, respectively. Hence, for Japanese one would actually only need one pattern for verbs in main clauses (`MainVerb`), and one pattern for verbs in relative clauses (`RelVerb`), respectively. The additional patterns given in table 11 are cases where we were not sure if the respective pattern subtype would really contribute positively to the lexicon, and which we hence wanted to test separately in our analysis step.
At first we allowed both entity labels to have arbitrary grammatical functions in our verb patterns; in particular, we did not require one of them to be in subject position.

The reasoning behind this was that since Japanese is a pro-drop language and may omit any verb argument — including the subject — in principle also lexicon entries for verbs in which both entities occupy non-subject positions may be turned into well-formed sentences. However, when looking at the entries generated by this first version of the patterns it turned out that gold lemmas occurred only in clauses where one of the entity labels occupies the subject position, and that other kinds of clauses most of the time do not express the desired relationship between the two entities, as illustrated by examples 28 and 29 below. Hence, in order to reduce noise at current all verb patterns only match clauses where the label of one of the entities from the triple store is most probably in subject position, i.e. where it is either marked by the subject particle が (*ga*) or the topic particle は (*wa*) without any preceding particles, which most of the time indicates that it is a substitute for the subject particle.
In contrast to the English, German and Spanish patterns for verb occurrences in the passive voice (Walter 2017, 131-136) the Japanese verb patterns also match clauses in the passive voice without an overt agent, i.e. clauses which when transferred into active voice would not have an overt subject, as exemplified by sentences 30 to 33 below: It turned out that such clauses regularly contained gold lemmas (30) or expressed the desired relation between the entities from the triple store by some other matching verbalization (32).

(28) *property:* `parent`

安楽公主-と共に	中宗-を	毒殺した
anrakukoushu-totomoni	*chuusou-o*	*dokusatsushita*
Princess.Anle-together.with	Emperor.Zhongzong.of.Tang-DOBJ	poisoned

[someone] poisoned Emperor Zhongzong of Tang together with Princess Anle

(29) *property:* `occupation`

会長職-を	李健熙-に	返上した
kaichoushoku-o	*igunhe-ni*	*henjoushita*
chairman.position-DOBJ	Lee.Kun.hee-IOBJ	gave.up

[someone] gave up the chairman position to Lee Kun-hee

(30) *property:* `yearOfConstruction`

グラニット鉄道-は、 1826年4月1日-に
guranittotetsudou-wa *senhappyakkunijuurokunenshigatsutsuitachi-ni*
Granite.railway-TOP 1826.year.4.month.1.day-on
着工された
chakkousareta
was.started

[the construction of] the Granite Railway was started on April 1, 1826

(31) グラニット鉄道-を 1826年4月1日-に
guranittotetsudou-o *senhappyakkunijuurokunenshigatsutsuitachi-ni*
Granite.railway-DOBJ 1826.year.4.month.1.day-on
着工した
chakkoushita
started

[someone] started [the construction of] the Granite Railway on April 1, 1826

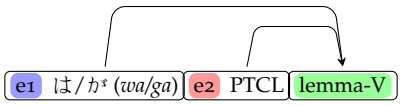
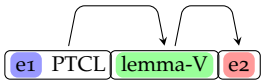
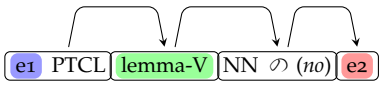
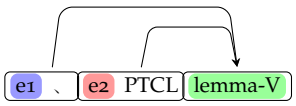
(32) *property:* `occupation`

声優-として 植田佳奈-が 採用された
seiyuu-toshite *uedakana-ga* *saiyousareta*
voice.actor-as Kana.Ueda-SUBJ was.employed

Kana Ueda was employed as a voice actor

(33) 声優-として 植田佳奈-を 採用した
seiyuu-toshite *uedakana-o* *saiyoushita*
voice.actor-as Kana.Ueda-DOBJ employed

[someone] employed Kana Ueda as a voice actor

Name	Dependency structure	Example	Can be merged with
`MainVerb`		ポモナ・ホール-は *pomonahooru-wa* Pomona.Hall-TOP 1726年-に *sennanahyakkunijuurokunen-ni* 1726.year-in 建てられました。 *tateraremashita* was.constructed Pomona Hall was constructed in 1726.	-
`RelVerb`		俳優-として 活動している *haiyuu-toshite* *katsudoushiteiru* actor-as have.been.active 伊勢谷友介 *iseyayuusuke* Yusuke.Iseya Yusuke Iseya, who has been active as an actor	-
`RelVerbNoAppos`		声優-を 務める *seiyuu-o* *tsutomeru* voice.actor-DOBJ serve スマイレージ-の 福田花音 *sumaireeji-no* *fukudakanon* S/mileage-POSS Kanon.Fukuda S/mileage's Kanon Fukuda, who serves as a voice actor	`RelVerb`
`MainVerbPtclEllipsis`		1610年、 *senroppyakkujuunen* 1610.year 総督邸-を *soutokutei-o* Governor's.house-DOBJ 建てた。 *tateta* built In 1610, [someone] built the Governor's house.	`MainVerb`

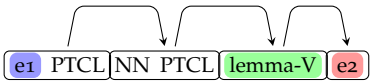
`RelVerbNestedE1`		1795年-から *sennanahyakkukyuujuugonen-kara* 1795.year-from *1798*年-に *sennanahyakkukyuujuuhachinen-ni* 1798.year-to 建設された *kensetsusareta* was.built 「マサチューセッツ州会議事堂」 *masachuusettsushuukaigijidou* Massachusetts.state.house the "Massachusetts state house", which was built from 1795 to 1798	`RelVerb`

Table 11: SPARQL queries for verb lemmas. The boxes indicate bunsetsu boundaries.

5.5 LEXICON ENTRY GENERATION

When a sentence matches one of the SPARQL queries, in order to create the actual lexicon entry M-ATOLL matches the output of the SPARQL query to one of several templates, which roughly correspond to the different (sub-)parts-of-speech a candidate verbalization may belong to and generate a lemon-based lexicon entry for the candidate verbalization at hand. As was already shown in the example in figure 14 the syntactic behavior of the candidate verbalization is defined in terms of one of the subcategorization frames specified in the linguistic ontology LexInfo (Cimiano et al. 2011), which describe the syntactic argument structure of candidate verbalizations.
As mentioned before, the noun patterns for Japanese are very similar to those defined for English, German and Spanish, and accordingly the already existing template `NounWithPrep` would in principle have been a rather good match for generating lexicon entries for Japanese candidate noun verbalizations. However, when this template is used, throughout the resulting lexicon entry the term *preposition* is used, as shown by the following English entry:

- canonical form: *discoverer*
- part-of-speech: common noun
- subcategorization frame: noun PP frame
  arguments:
  - copulative argument e_1
  - prepositional object e_2 with preposition *of*
- semantic reference: `discoverer`
  arguments:
  - subject e_1
  - object e_2

Since Japanese particles are not pre- but postpositions, this terminology would be unfavorable in Japanese lexicon entries. Hence, we defined a kind of more general template `NounWithAdpos` that only differs from `NounWithPrep` in that it references adpositions instead of prepositions:

- canonical form: 発見者 (*hakkensha*)

- part-of-speech: common noun
- subcategorization frame: noun AdP frame
  arguments:
  - copulative argument e_1
  - adpositional object e_2 with adposition の (*no*)
- semantic reference: `discoverer`
  arguments:
  - subject e_1
  - object e_2

Since in Japanese all verb arguments are marked the same way, in contrast to English, German and Spanish we do not differentiate between templates for transitive and intransitive verbs with an adpositional argument. Rather, we make use of two new templates, `ActiveVerb` and `PassiveVerb`, that each create lexicon entries which reference a subcategorization frame with a subject and an adpositional object. For example, for sentence 30 the `PassiveVerb` template would be invoked and the following entry would be created:

- canonical form: 採用される (*saiyousareru*)
- part-of-speech: verb
- subcategorization frame: passive AdP frame
  arguments:
  - subject e_1
  - adpositional object e_2 with adposition として (*toshite*)
- semantic reference: `occupation`
  arguments:
  - subject e_1
  - object e_2

Here, transitive and intransitive verbs only differ in that for transitive verbs the marker of the adpositional object is always を (*o*), while for intransitive verbs it is any other marker. While it would be possible to use only one single template for all Japanese verb

lemmas by turning the passive verbs into their active form, in cases such as example 30 or 32 this would lead to entries without a subject.

5.6 EVALUATION

5.6.1 *SPARQL Queries*

In order to check how comprehensive our SPARQL queries for dependency patterns are, we took the gold lexicon for the properties we used to generate the SPARQL queries and looked at how many instances of the lemmas from this gold lexicon are found by our queries, and how many gold instances are occurring overall in positions where they may in principle express a relationship between subjects and objects from DBpedia's triple store. In order to determine the latter value, for each example property we retrieved all sentences containing labels of elements linked in DBpedia's triple store by the respective property, and counted how often the gold lemmas for the property at hand occurred between or behind these triple subjects and objects. Since Japanese is a strongly head-final language, this should cover all instances of the gold lemmas that may potentially express a relation between triple subjects and objects. The results are shown in table 12, together with counts of how often each gold lemma was actually found by our SPARQL queries.

property	# sent.s	verbalization	# instances of lemma between/after triple subject and object	# sent.s found through SPARQL queries	% coverage
`crosses`	241	跨ぐ (*matagu*; 'to step over, to bridge')	5	2	40
		架かる (*kakaru*; 'to span, to cross')	91	12	13.19
		かかる (*kakaru*; writing variant of 架かる)	17	2	11.76
		またがる (*matagaru*; 'to extend over')	1	1	100
		渡る (*wataru*; 'to cross over')	20	2	10
`nationality`	4660	国籍 (*kokuseki*; 'nationality')	9	2	22.22
		出身 (*shusshin*; 'person's origin')	283	38	13.43
		生まれ (*umare*; 'birthplace')	33	3	9.09
`occupation`	9531	仕事 (*shigoto*; 'work')	58	0	0

		職業 (*shokugyou*; 'occupation')	10	0	0
		職 (*shoku*; 'job, position')	16	0	0
		生業 (*nariwai*; 'job')	0	0	-
		勤める (*tsutomeru*; 'to work (for)')	9	1	11.11
		務める (*tsutomeru*; 'to serve (as)')	342	16	4.68
		活動 (*katsudou*; 'activity')	311	5	1.61
		働く (*hataraku*; 'to work')	9	0	0
`yearOfConstruction`	281	完成 (*kansei*; 'completion')	11	4	36.36
		竣工 (*shunkou*; 'completion of construction')	2	0	0
		建設 (*kensetsu*; 'construction')	16	3	18.75
		建てる (*tateru*; 'to build')	5	3	60
`parent`	11,831	子供 (*kodomo*; 'child')	104	2	1.92
		子 (*kou*; 'child [of someone]')	948	31	3.27
		父親 (*chichioya*; 'father [of someone]')	54	3	5.56
		父 (*chichi*; 'father')	651	69	10.60
		娘 (*musume*; 'daughter')	928	65	7.00
		息子 (*musuko*; 'son')	1457	179	12.29
		親 (*oya*; 'parent')	9	0	0
		母 (*haha*; 'mother')	446	17	3.81
		母親 (*hahaoya*; 'mother [of someone]')	49	1	2.04

Table 12: Number of instances of gold lemmas found between or after triple subjects and objects, and number of instances that are actually found by our SPARQL queries

Out of the 29 lemmas in the gold lexicon, seven were not found at all by the SPARQL queries, which corresponds to a recall of 0.76. In only one case this is due to the lemma

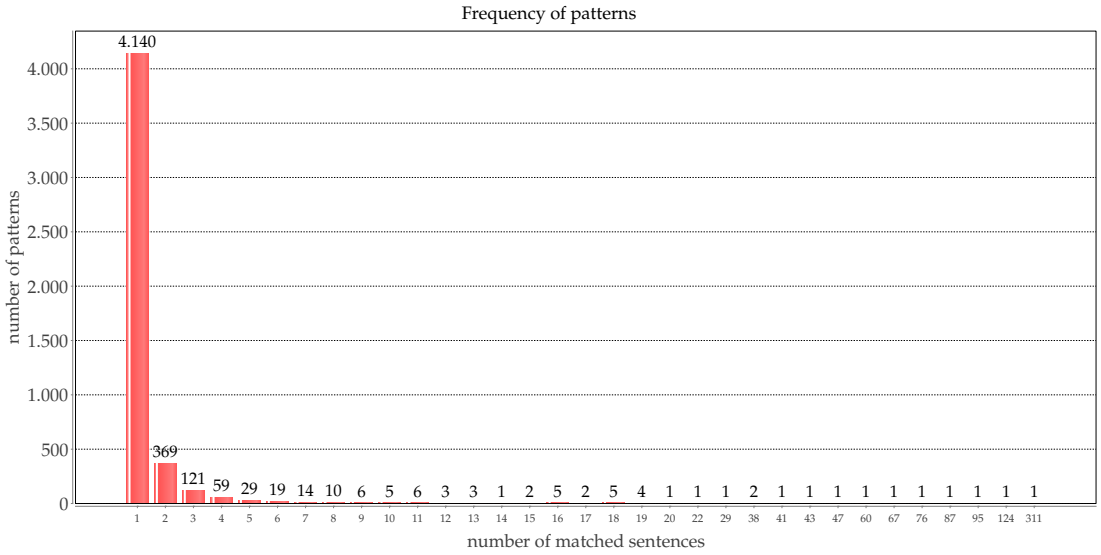
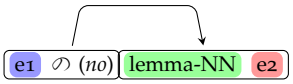
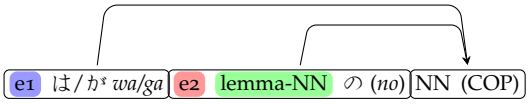
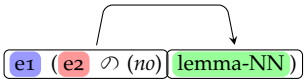
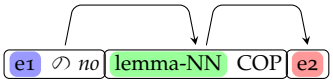
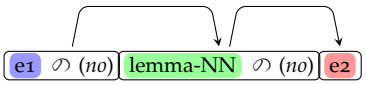
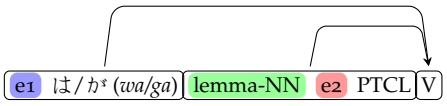


Figure 18: Frequency of dependency patterns

not occurring between or behind triple subjects and objects at all. Furthermore, the overall coverage of instances of the gold lemmas by the SPARQL queries was rather low: For example, for a lemma such as 務める (*tsutomeru*; 'to serve') only 16 instances are found by SPARQL queries, while over 300 occur between or behind triple subjects and objects. As the number of instances found for a given lemma may serve as an important parameter when deciding which generated entries to keep in the lexicon and which to discard, we first looked into how to to improve the overall coverage of our SPARQL queries in terms of found instances, and whether in the process the recall, i.e. the coverage in terms of found lemmas, may improve as well.

For each instance of a gold lemma found between or behind a triple subject and object, we constructed the minimal path between these three elements within the sentence's dependency tree, and grouped together all instances that share the same path structure. As is shown in figure 18, the vast majority of dependency patterns that show up this way occurs only once. Basing new SPARQL queries or modifications to existing queries on patterns that only occur a few times overall would probably not be very worthwhile. Therefore, we looked at the ten dependency patterns occurring most frequently in more detail, checking whether they were already covered by our SPARQL queries , and if not, whether it would make sense to build new SPARQL queries based on them, the results of which are shown in table 13.

Pattern	example	# sent.s	SPARQL pattern?
	フリードリヒ4世-の 息子 *furiidorihiyonsei-no musuko* Friedrich.IV-POSS son フリードリヒ5世 *furiidorihigosei* Friedrich.V Friedrich IV's son Friedrich V	387	yes (`JA_Apposition`)
	アン・ヒューズ-は イギリス *anhyuuzu-wa igirisu* Ann.Hughes-TOP England 出身-の 柔道選手。 *shusshin-no juudousenshu* origin-POSS judo.player Ann Hughes is a judo player of English origin.	128	yes (`JA_MainCopulaAppos`)
	スレイマン1世 *sureimanissei* Suleiman.the.Magnificent （セリム1世-の 子） *serimuissei-no kou* Selim.I-POSS child Suleiman the Magnificent (Selim I's child)	124	no (new pattern `JA_Appos-Brackets` created)
	リチャード1世-の 父親 *richaadoissei-no chichioya* Richard.I-POSS father である ヘンリー2世 *dearu henriinisei* COP Henry.II Henry II, who is the father of Richard I	95	yes (`JA_RelCopulaCompl`)
	リュクルゴス-の 子-の *ryukurugosu-no kou-no* Lykurgos-POSS child-ADN ペロプス *peropusu* Pelops Lykurgo's child Pelops	67	no (new pattern `JA_DoubleNo` created)

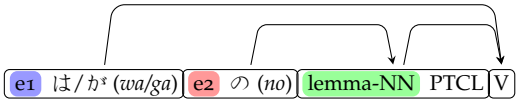


ウァレリアヌス-は 息子
uarerianusu-wa *musuko*
Valerian-TOP son
ガッリエヌス-を
garrienusu-o
Galienus-DOBJ
ローマ帝国-の
roomateikoku-no
Roman.Empire-POSS
西半分-を 任せた
nishihanbun-o *makaseta*
western.half-DOBJ left

Valerian left the western half of the Roman Empire to [his] son Galienus.

60

no (too specific to `parent`?)

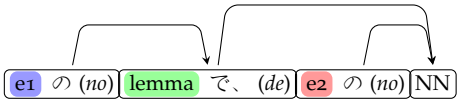


ラージャーラーム-は
raajaaraamu-wa
Rajaram-TOP
シヴァージー-の 息子-として
shivuaajii-no *musuko-toshite*
Shivay-POSS son-as
生まれた
umareta
was.born

Rajaram was born as the son of Shivay

48

no (in ca. 50% of cases no relationship between e1 and e2 is expressed)



ロマノス2世-の 娘
romanosunisei-no *musume-de*
Romanos.II-POSS daughter
で、
bashireiosunisei-no
COP
バシレイオス2世-の 妹。
imouto
Basilius.II-POSS sister

[She] is the daughter of Romanos II and the sister of Basilius II.

47

no (expresses no direct relationship between e1 and e2)

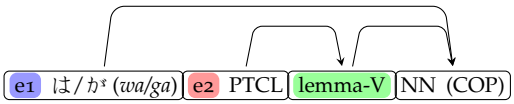
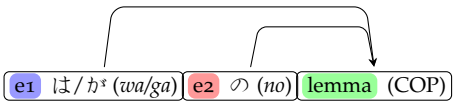
	シャンジュ橋-は、 *shanjuhashi-wa* Change.bridge-TOP セーヌ川-に 架かる 橋 *seenugawa-ni* *kakaru* *hashi* Seine-IOBJ to.cross bridge である。 *dearu* COP Pont au Change is a bridge that crosses the Seine.	38	no (new pattern JA_RelVerbMain-Copula created)
	クレオメネス3世-は *kureomenesusansei-wa* Cleomenes.III-TOP レオニダス2世-の 息子 *reonidasunisei-no* *musuko* Leonidas.II-POSS son である。 *dearu* COP Cleomenes III is the son of Leonidas II.	37	yes (JA_Main-CopulaCompl)

Table 13: Most frequently occurring patterns over all lemmas from gold lexicon

Four out of these ten dependency patterns were already covered by SPARQL queries. In addition, we wrote three more queries for patterns from the list that on the one hand seemed not too specific to a certain property and in which on the other hand the lemma seems to actually express a relationship between the triple subject and object in the majority of cases; the latter was decided based on a sample of ten random instances of the respective pattern. The remaining three patterns were considered unsuitable for being turned into SPARQL queries: In one pattern the lemma does not express a relationship between the triple subject and object, but between the triple subject and another noun; in a further case, the pattern seems to convey a relationship between subject and object in only around half of all instances, and incorporating this pattern as a SPARQL query would hence most likely result in lots of incorrect entries. Finally, in the third case we suspected the pattern may be very specific to the property `parent`, and may produce lots of erroneous data for other properties. It should be noted that in addition to the patterns occurring most frequently in total, we also looked at the most frequent dependency patterns over those sentences that M-ATOLL did not cover yet. This way, it turned out that a number of sentences that the already existing SPARQL

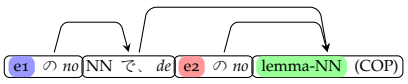
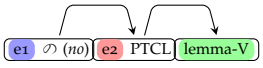
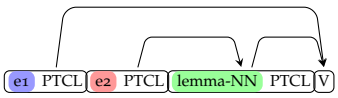
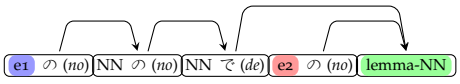
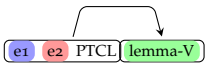
queries were supposed to match were not found yet, and according modifications were applied to the queries to improve their coverage.
Table 14 shows how these modifications and the introduction of the three new SPARQL queries influence the number of lemma instances found by M-ATOLL.

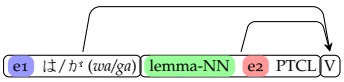
property	verbalization	# instances of lemma between/after triple subject and object	# sent.s found through SPARQL queries		% coverage	
			before analysis	after analysis	before	after
`crosses`	跨ぐ (*matagu*; 'to step over, to bridge')	5	2	2	40	40
	架かる (*kakaru*; 'to span, to cross')	91	12	47	13.19	51.65
	かかる (*kakaru*; writing variant of 架かる)	17	2	4	11.76	23.53
	またがる (*matagaru*; 'to extend over')	1	1	1	100	100
	渡る (*wataru*; 'to cross over')	20	2	3	10	15
`nationality`	国籍 (*kokuseki*; 'nationality')	9	2	2	22.22	22.22
	出身 (*shusshin*; 'person's origin')	283	38	119	13.43	42.05
	生まれ (*umare*; 'birthplace')	33	3	6	9.09	18.18
`occupation`	仕事 (*shigoto*; 'work')	58	0	0	0	0
	職業 (*shokugyou*; 'occupation')	10	0	0	0	0
	職 (*shoku*; 'job, position')	16	0	0	0	0
	生業 (*nariwai*; 'job')	0	0	0	-	-
	勤める (*tsutomeru*; 'to work (for)')	9	1	1	11.11	11.11
	務める (*tsutomeru*; 'to serve (as)')	342	16	40	4.68	11.70
	活動 (*katsudou*; 'activity')	311	5	6	1.61	1.93
	働く (*hataraku*; 'to work')	9	0	0	0	0
`yearOfConstruction`	完成 (*kansei*; 'completion')	11	4	5	36.36	45.45
	竣工 (*shunkou*; 'completion of construction')	2	0	1	0	50
	建設 (*kensetsu*; 'construction')	16	3	3	18.75	18.75
	建てる (*tateru*; 'to build')	5	3	3	60	60
`parent`	子供 (*kodomo*; 'child')	104	2	36	1.92	34.62
	子 (*kou*; 'child [of someone]')	948	31	102	3.27	10.76
	父親 (*chichioya*; 'father [of someone]')	54	3	8	5.56	14.81
	父 (*chichi*; 'father')	651	69	106	10.60	16.28

娘 (*musume*; 'daughter')	928	65	102	7.00	10.99
息子 (*musuko*; 'son')	1457	179	337	12.29	23.13
親 (*oya*; 'parent')	9	0	0	0	0
母 (*haha*; 'mother')	446	17	24	3.81	5.38
母親 (*hahaoya*; 'mother [of someone]')	49	1	2	2.04	4.08

Table 14: Number of instances of gold lemmas found between or after triple subjects and objects, and number of instances that are actually found by our SPARQL queries, after new patterns found through analysis of minimal dependency paths have been added

Overall, at least for some lemmas significantly more instances are found now; however, the recall has improved only very slightly: Only one further lemma is found, in only one sentence. In general, it seems that the applied changes lead to lemmas which were found frequently already before to be found even more often, while for lemmas which were found only a few times — or not at all — the numbers did not change much. In order to check if we could also improve coverage and recall for the less frequently found lemmas, we again looked at a list of most frequently occurring dependency patterns, this time based only on sentences not found by M-ATOLL yet and and a reduced set of gold lemmas with the five most frequently found ones being removed. As can be seen from the results in table 15, this time the found patterns were of significantly lower quality: In most cases none of the sentences belonging to a given pattern express a direct relationship between triple subject and object — at least not by means of the lemma at hand — and one further pattern which already occurred in table 13 seems too specific to the property `parent`. Since any further dependency pattern occurring in the data would at most match three instances of less frequently covered lemmas, we decided that looking at further patterns would probably not be worthwhile and did not apply any further changes to our set of SPARQL queries.

Pattern	example	# sent.s	comment
	キャサリン・オブ・ヴァロワ-は、 *kyasarinobuvuarowa-wa* Catherine.of.Valois-TOP ヘンリー5世-の 王妃 で、 *henriigosei-no ouhi de* Henry.V-POSS queen COP ヘンリー6世-の 母。 *henriirokusei-no haha* Henry.VI-POSS mother Catherine of Valois is Henry V's queen and the mother of Henry VI.	20	expresses no (direct) relation between e1 and e2
	音楽プロデュース-は 作曲家-の *ongakupurodeyuusu-wa sakkyokuka-no* music.producer-TOP composer-ADN 澤野弘之-が 務めている。 *sawanohiroyuki-ga tsutometeiru* Hiroyuki.Sawano-SUBJ is.serving Composer Hiroyuki Sawano is serving as the music producer.	23	relation between e1 and e2 is not expressed by lemma, but by adnominalizer の (*no*)
	ラージャーラーム-は 3月-に *raajaaraamu-wa sangatsu-ni* Rajaram-TOP March-at 死亡しており、 シヴァージー2世-と その *shiboushiteori shivuaajiinisei-to sono* die Shivaji.II-COM that 母 ターラー・バーイー-は 逃げた。 *haha taaraabaaii-wa nigeta* mother Thaler.Bai-TOP fled Rajaram died in March, and Shivaji II and his mother Thaler Bai fled.	8	none of the sentences expresses relationship between e1 and e2 (by means of lemma)
	ファールーク1世王-の 2番目-の *faaruukuisseiou-no nibanme-no* Farouk.I.king-POSS second-ADN 妻 で、 フアード2世-の 母。 *tsuma de fuaadonisei-no haha* wife COP Fuad.II-POSS mother [She] is the second wife of Farouk of Egypt and the mother of Fuad II.	6	similar to second pattern above; expresses no (direct) relation between e1 and e2
	声優・ 川上とも子-が *seiyuu kawakamitomoko-ga* Voice.actor Tomoko.Kawakami-SUBJ パーソナリティ-を 務めていた *paasonariti-o tsutometeita* personality-DOBJ have.served インターネットラジオ番組。 *intaanettorajiobangumi* internet.radio.program [It is] the internet radio program on which voice actor Tomoko Kawakami has served as a [radio] personality.	6	Similar to first pattern above; relationship between e1 and e2 is not expressed through lemma



ウァレリアヌス-は *uarerianusu-wa* Valerian-TOP ガッリエヌス-を *garrienusu-o* Galienus-DOBJ 西半分-を *nishihanbun-o* western.half-DOBJ	息子 *musuko* son ローマ帝国-の *roomateikoku-no* Roman.Empire-POSS 任せた *makaseta* left	4	none (yet) (too specific to `parent`?)

Valerian left the western half of the Roman Empire to [his] son Galienus.

Table 15: Patterns not covered yet over lemmas from gold lexicon without the five lemmas most frequently found by current set of SPARQL patterns (息子 [*musuko*; found 337 times], 出身 [*shusshin*; 119], 父 [*chichi*; 106], 娘 [*musume*; 102], 子 [*kou*; 102]), with at least four matches

Finally, we wanted to find out which impact each single SPARQL query has on the quality of the overall lexicon generated by M-ATOLL, the results of which are shown in figures 19 and 20. Both for the properties used in this section (figure 19) and a set of five new properties used in the analyses in the following section (figure 20), we computed precision, recall and f-measure for two different sets of lexicons with respect to two handwritten gold lexicons for the respective sets of properties. On the one hand, we generated a set of lexicons where for each lexicon all but one distinct SPARQL query were used in the lexicon generation process, which allowed us to compute how precision, recall and f-measure are affected when this one pattern is left out, as is shown in the upper rows of figures 19 and 20, respectively. If recall is lower for one of these lexicons than for the lexicon with all patterns being used, the pattern left out in the lexicon at hand matches verbalizations from the gold lexicon that are not matched by any other pattern. In contrast, if recall is the same for the lexicon at hand and the lexicon covering all patterns, the respective pattern does not add any gold entries that are not matched by some other pattern. Both for the set of old and new properties only very few — three and two, respectively — patterns seem to add gold lemmas that are not matched by any other pattern, with only one of these patterns — `RelVerb` — behaving like this for both sets of properties. At least for the old properties this one pattern seems to be particularly relevant, in that without it recall drops by around 19 percentage points, which may serve to show how important SPARQL patterns for relative clauses are at least in case of Japanese. The fact that recall does not drop when a given pattern is left out may either mean that it does not match any gold verbalizations at all, in which case it may just as well be removed from the set of SPARQL patterns, or that it only matches gold verbalizations that are also matched by other patterns. The

latter would not necessarily be a reason to remove the pattern, as depending on the strategy one employs for further filtering the lexicon entries, having several matches of a given suitable verbalization in one's lexicon may actually be an advantage.
In order to find out more about how each SPARQL query affects the quality of the overall lexicon, we also looked at a further set of lexicons that were generated with only one single query being used, respectively, as shown in the lower rows of figures 19 and 20, respectively. One query, `MainDobjPtclEllipsis`, has a recall of zero for both sets of properties and was therefore left out in the remaining analyses carried out in the following section. Again, one can tell from the recall that the `RelVerb` pattern seems to have a special position in that out of all patterns it seems to add the largest number of gold verbalizations to the lexicon, in case of the new set of properties together with another relative clause pattern, `RelVerbMainCopula`. Out of the seven patterns that were detected by means of our semi-automatic approach to the generation of dependency patterns, four — `JA_Apposition`, `JA_MainCopulaAppos`, `JA_ApposBrackets` and `JA RelCopulaCompl` — seem to have a significant positive effect on either of the two sets of lexicons, in that either without them the recall drops, or the lexicons that were generated with them as the only active dependency pattern have an above average recall.
Overall, precision is very low. Two factors that may contribute to this are that on the one hand, we have not yet applied any filtering to the entries generated by M-ATOLL, and that on the other hand, the handwritten gold lexicons for both the old and the new sets of properties are rather small, with a total of 33 or 26 entries, respectively. How both of these points relate to the precision of our lexicons will be looked at in the next section.

5.6.2 *Verbalizations Retrieved by M-ATOLL*

In order to test how well the SPARQL patterns generalize, we computed precision, recall and f-measure for the ontology lexicon generated by M-ATOLL on the properties used already in the preceding section, and compared these values to those of another M-ATOLL lexicon generated for five new properties, `author`, `bandMember`, `foundingYear`, `languageFamily`, and `locationCity`. The results for the old set of example properties are shown in figure 21, while the results for the five new example properties are depicted in figure 22. As mentioned before, the entries created by M-ATOLL should be filtered in some way; we looked at two different filtering strategies, both based on

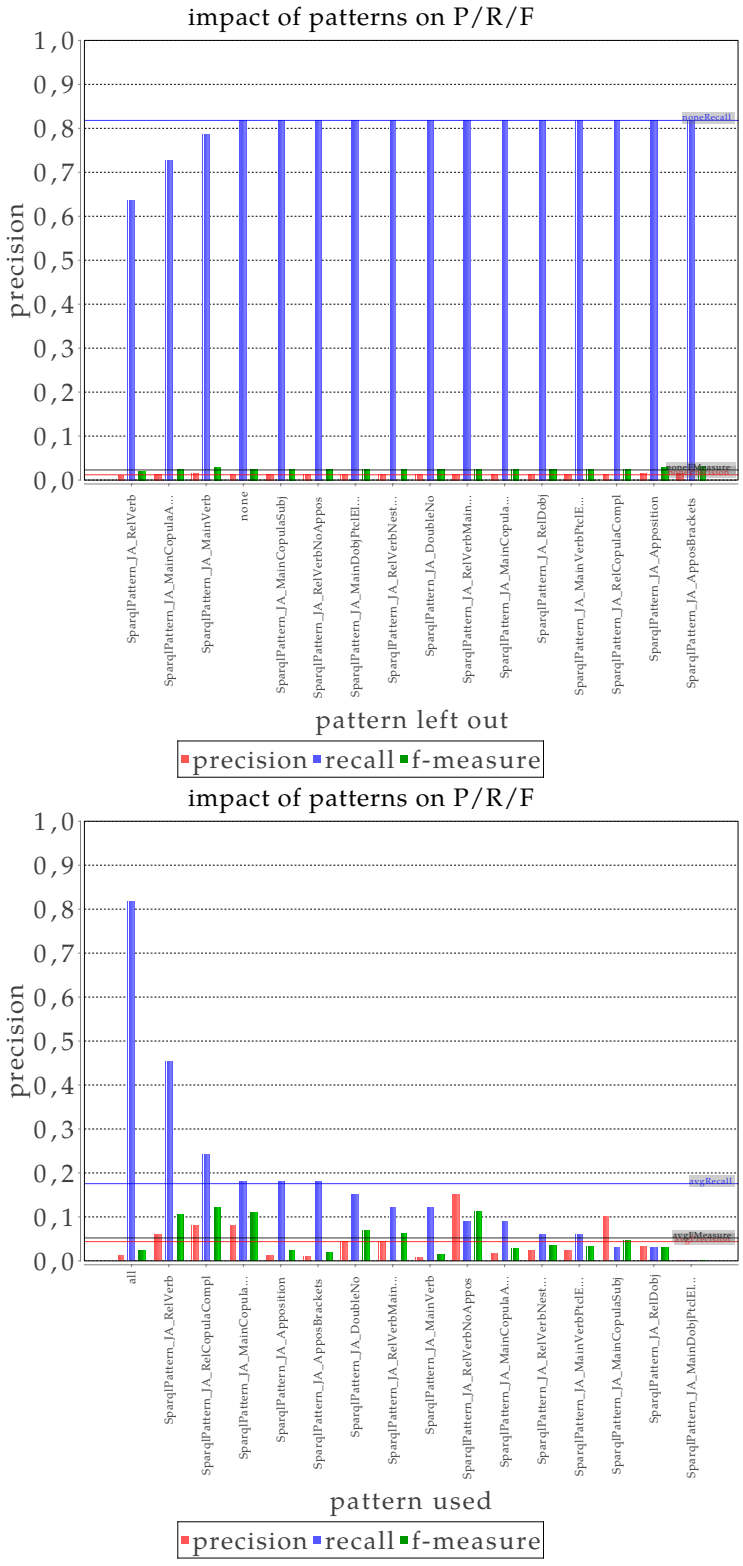


Figure 19: Precision, recall and f-measure for lexicons based on the old set of properties. Upper row: Measures for lexicons in whose generation one specific SPARQL pattern was left out, respectively. Lower row: Measures for lexicons for whose generation only one specific SPARQL pattern was used, respectively.

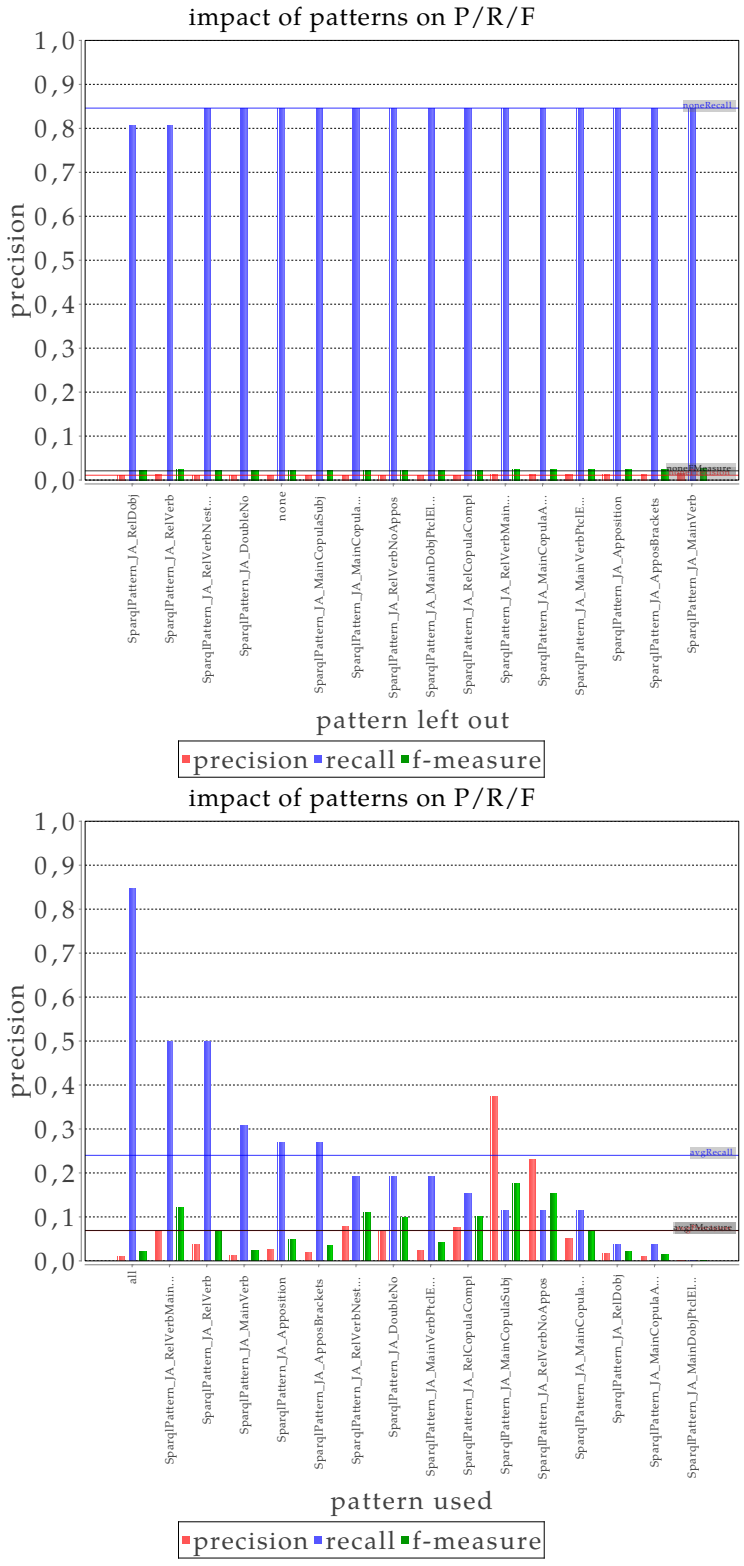


Figure 20: Precision, recall and f-measure for lexicons based on a set of five new properties. Upper row: Measures for lexicons in whose generation one specific SPARQL pattern was left out, respectively. Lower row: Measures for lexicons for whose generation only one specific SPARQL pattern was used, respectively.

the number of times a given lemma has been found in the corpus: On the one hand, we chose all entries for inclusion in the final lexicon whose lemma appeared at least a certain number of times in the corpus, the results of which are shown in the upper rows of figures 21 and 22, respectively. For example, the lexicon whose values are listed on point four of the x-axis contains all entries whose respective lemma appears four or more times in the corpus. On the other hand, we sorted the entries according to the number of occurrences of their lemmas in descending order, and included only the first x entries from this list in the final lexicon, the results of which can be seen in the lower rows of figures 21 and 22. In this case, the lexicon whose values are given at point four of the x-axis would contain the entries for the four most frequently occurring lemmas. One should note that since multiple lemmas may occur the same amount of times, the sorting of the entries may not be definite, and different entries may show up in the final lexicon if the filtering process is repeated. This second filtering strategy may be of advantage if one always wants to have a certain, fixed number of entries in one's final lexicon. Furthermore, it may be preferable when the number of entries M-ATOLL is able to extract differs significantly among different properties: For a property for which only a few entries are extracted, lemmas with only one or two occurrences may already be good verbalizations, while for a property with hundreds or thousands of generated entries such lemmas would most probably not be suitable.

As can be seen from figures 21 and 22, the measures are roughly comparable among both lexicons. For smaller lexicons precision is higher for the lexicons based on the old properties, while for larger lexicons recall is slightly higher for the lexicons based on the new properties. However, overall the numbers seem to suggest that the SPARQL queries used for Japanese by M-ATOLL are not overfitted to the properties we used in section 5.6.1. While the recall of both the lexicon for the old and new properties can be brought to a halfway acceptable level with the right filtering strategy, precision is overall very low, i.e. only few of the entries in the M-ATOLL lexicons correspond to entries from the manually created gold lexicons. Therefore, for the M-ATOLL lexicon covering the new properties we looked at the top 20 entries received by the second filtering strategy described above, and checked whether these entries are really of low quality for the most part, or whether there may be some problem with the gold lexicon in terms of coverage instead. The original entries of the gold lexicon, plus additional entries from the top 20 lexicon we considered appropriate for verbalizing the respective property, are shown in table 16. For every property there were at least five such additional entries. For the most part they were not included in the gold lexicon due to

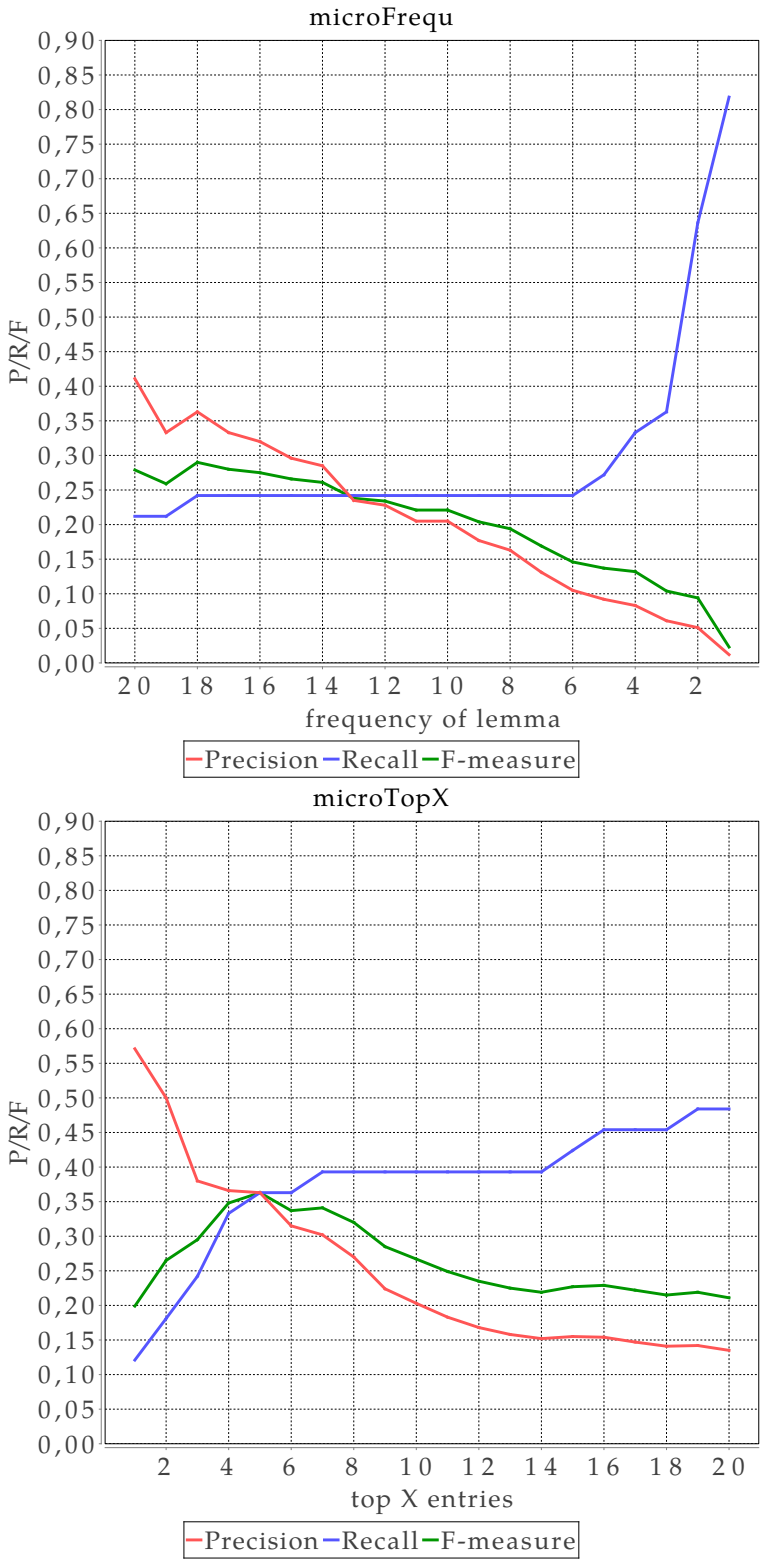


Figure 21: Precision, recall and f-measure for the lexicons generated by M-ATOLL for the old example properties; entries are filtered based on the number of occurrences of their lemma (upper row) or based on where in a list sorted according to the number of lemma occurrences they occur (lower row)

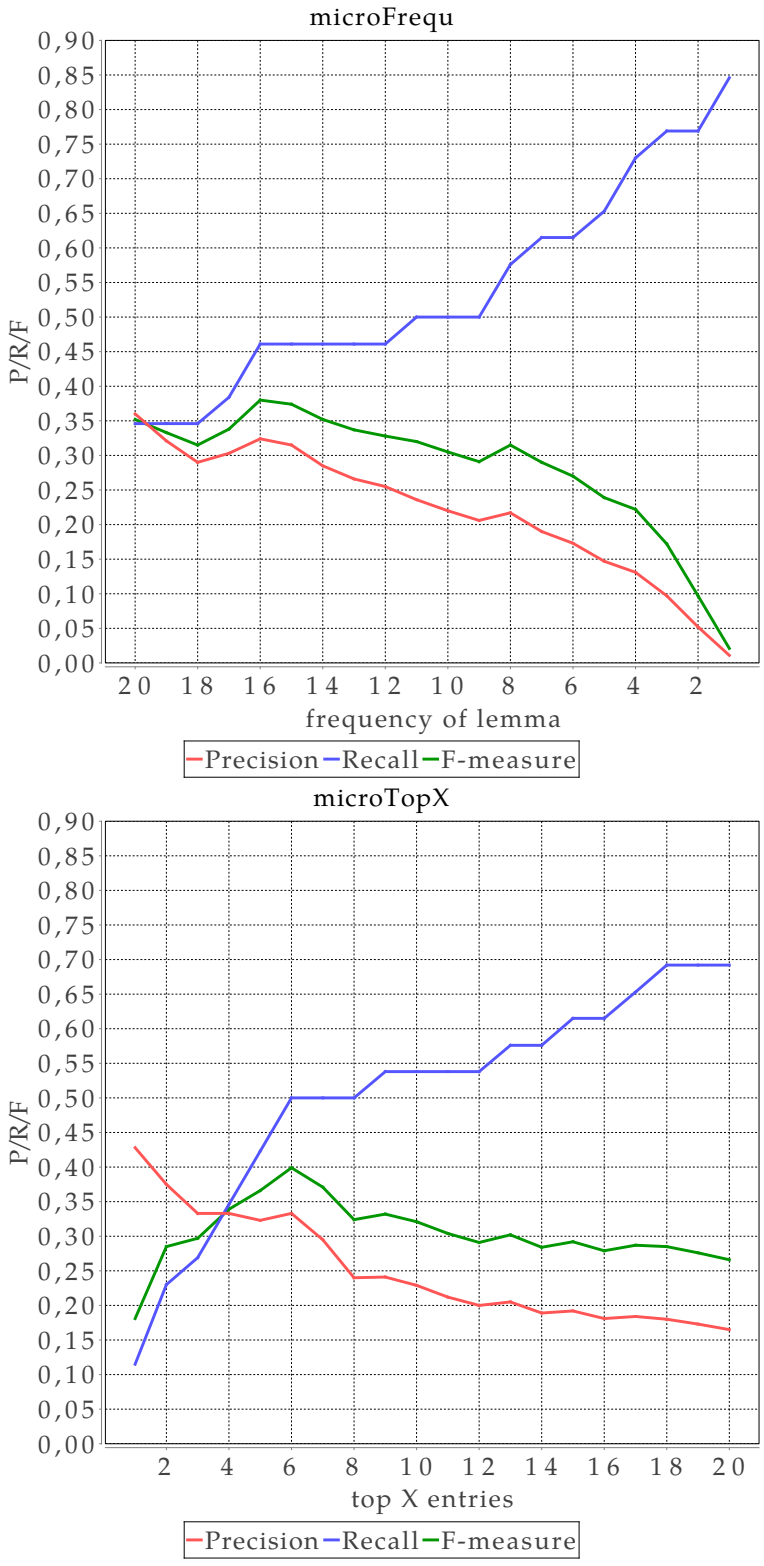


Figure 22: Precision, recall and f-measure for the lexicons generated by M-ATOLL for five new example properties; entries are filtered based on the number of occurrences of their lemma (upper row) or based on where in a list sorted according to the number of lemma occurrences they occur (lower row)

property	*original lemmas*	*additional lemmas found in top 20 entries of lexicon generated by M-ATOLL*
`author`	著者 (*chosha*; 'writer') 書く (*kaku*; 'to write') 作家 (*sakka*; 'novelist') 著作家 (*chosakuka*; 'author') 作品 (*sakuhin*; 'work, opus') 作 (*saku*; 'work [of art]') 作者 (*sakusha*; 'author')	著す (*arawasu*; 'to write [a book]') 漫画 (*manga*; 'manga') 小説 (*shousetsu*; 'novel') 執筆 (*shippitsu*; 'writing [as a profession]') 原作者 (*gensakusha*; 'original author')
`bandMember`	バンドメンバー (*bandomenbaa*; 'band member') メンバー (*menbaa*; 'member') 所属 (*shozoku*; 'belonging to [used for humans]')	ギタリスト (*gitarisuto*; 'guitarist') ボーカリスト (*bookarisuto*; 'vocalist') ボーカル (*bookaru*; abbrev. of 'vocalist') リーダー (*riidaa*; 'leader') 結成 (*kessei*; 'forming [a group of people, e.g. band, team]') ヴォーカル (*vuookaru*; altern. writing form of abbrev. of 'vocalist') 音楽ユニット (*ongakuyunitto*; "music unit"; certain type of J-Pop band) ベーシスト (*beeshisuto*; 'bassist') ユニット (*yunitto*; "unit") 音楽ユニットギタリスト (*ongakuyunittogitarisuto*; 'music unit guitarist') ロックバンド (*rokkubando*; 'rock band')
`foundingYear`	設立 (*setsuritsu*; 'founding') 創立 (*souritsu*; 'establishment') 創設 (*sousetsu*; 'founding') 創始 (*soushi*; 'creation') 成立 (*seiritsu*; 'coming into existence')	組織 (*soshiki*; 'organization, construction') 発足 (*hossoku*; 'start') 建国 (*kenkoku*; 'founding of a nation') 創業 (*sougyou*; 'establishment [of a business]') 独立 (*dokuritsu*; 'becoming independent') 始まる (*hajimeru*; 'to start')
`languageFamily`	属す (*zokusu*; 'to belong to') 言語 (*gengo*; 'language') 含む (*fukumu*; 'to include')	一種 (*isshu*; 'one kind, variety') 一つ (*hitotsu*; 'one [of several]') 分類 (*bunrui*; 'classification') ひとつ (*hitotsu*; writing variant of 'one') 種 (*kusa*; 'kind, variety')
`locationCity`	所在する (*shozaisuru*; 'to be located') 都市 (*toshi*; 'city') 場所 (*basho*; 'location')	置く (*oku*; 'to put, to place') 本社 (*honsha*; 'head office') 存在する (*sonzaisuru*; 'to exist') 設立 (*setsuritsu*; 'founding') 会社 (*kaisha*; 'company, corporation') 本拠地 (*honkyochi*; 'headquarters') 行う (*okonau*; 'to perform, to take place') 創業 (*sougyou*; 'establishment [of a business]') 構える (*kamaeru*; 'to set up') 一つ (*hitotsu*; 'one [of several]')

Table 16: Entries of gold lexicon for new properties

a mismatch in semantic granularity: Some of these verbalizations are more specific than the property at hand, such as ボーカリスト (*bookarisuto*; 'vocalist') as a verbalization of `bandMember`, while in other cases they are considerably more general, such as 一つ (*hitotsu*; 'one [of several]') as a verbalization of `languageFamily` or `locationCity`. Whether or not one would consider such verbalizations appropriate for being included in the final lexicon decidedly depends upon the application area at hand: In case of natural language generation, when the system needs to know which verbalizations it can use in its output, verbalizations more specific than the property at hand may lead to erroneous output, such as 'Don Quixote is a manga by Cervantes', at least if no further information about the semantics of those lemmas is provided in their respective entry. In contrast, more general terms, such as 一つ (*hitotsu*; 'one [of several]') in スペイン語はロマンス諸語の一つです (*supeingowa romansushogono hitotsu desu*) 'Spanish is one of the Romance languages', would work for this application area. Conversely, in case of natural language understanding, where the system needs to figure out if a given natural language input contains a reference to a given property, more specific verbalizations would be acceptable. For example, if the system received an input of the form *Don Quixote is a novel by Cervantes*, and no other properties apart from `author` are linked to that verbalization in the lexicon, it could be sure that the `author` property is expressed in that sentence. However, very general terms such as 一つ (*hitotsu*; 'one [of several]'), which would tend to be linked to a larger number of different properties, may lead to the system choosing the wrong property.

Apart from whether verbalizations whose semantics are not an exact match of the semantics of the property at hand should be included in the lexicon, a further question would be how such verbalizations could be represented appropriately. Verbalizations that are more specific may in many cases be modeled by restricting the range or domain of the verbalization's sense by means of lemon's so-called `propertyDomain` and `propertyRange` conditions, which serve to express that the subjects and/or objects that can occur with the given verbalization need to belong to a further — usually more specific — class than the one already specified for the subject or object by the referenced property. For example, in case of the `bandMember` property and the candidate verbalization ボーカリスト (*bookarisuto*; 'vocalist') given in table 16, a corresponding lexicon entry may look as follows:

```
:ボーカリスト a lemon:Word ;
[...]
        lemon:sense :ボーカリスト_sense .
```

```
[...]
:ボーカリスト_sense
        lemon:reference <http://dbpedia.org/ontology/bandMember> ;
        lemon:propertyRange <http://dbpedia.org/ontology/Singer> ;
        lemon:subjOfProp :ssyn ;
        lemon:objOfProp :osyn .
```

Determining automatically that a verbalization is more specific in this way would require that a system such as M-ATOLL is able to detect that the subjects and/or objects that occur with a given candidate verbalization always belong to a certain class other than the standard domain/range classes of the given property. For more general verbalizations there are several options for representing them in the lexicon: On the one hand, such verbalizations could be represented as referencing a union of several properties, which is feasible if it is known exactly which properties can all be referenced by the verbalization at hand. On the other hand, if this knowledge is not available, one may define a new condition that can be imposed upon the verbalization's sense as follows:

```
:作品 a lemon:Word ;
[...]
        lemon:sense :作品_sense .
[...]
:作品_sense lemon:reference <http://dbpedia.org/ontology/author> ;
        :coverage "more general" ;
        lemon:subjOfProp :ssyn ;
        lemon:objOfProp :osyn .

:coverage rdfs:subPropertyOf lemon:condition
```

Detecting such more general verbalizations automatically would require the system to discover that the candidate verbalization at hand occurs for more than one property. Apart from being more specific or more general, verbalizations may also in some other way be related to the property at hand. For example, in case of the `bandMember` property the verbalization 脱退する (を) (*dattaisuru (o)*; 'to retire, withdraw from') does not reference the property itself, but rather the end of the relationship that is expressed through this property. Expressing that a verbalization is related to a given property in this or a similar way may be done through defining a new condition, similarly to the strategy for more general verbalizations depicted above. However, detecting such related verbalizations automatically would at least be very difficult, and could most

probably only be done by hand.
Table 17 shows all candidate verbalizations from the top 20 entries retrieved as described above, sorted according to how their semantics relate to those of the property they are supposed to express, i.e. according to whether their semantics are roughly the same as, considerably more general or more specific than the semantics of the property at hand, whether their meaning is in some other way related to that of the property or whether their semantics are completely unrelated. For each property at most five candidate verbalizations fall under the last category, i.e. for every property at least 75% of the top 20 entries can be considered more or less suitable verbalizations. Overall, most additional verbalizations are more general than the respective property; however, which category most verbalizations belong to differs considerably among the properties: For `bandMember`, for example, half of the top 20 entries are more specific than the property, while for `languageFamily` over half of the candidate verbalizations are considerably more general.

property	verbalizations
`author`	*direct lexicalization*: 5
	作者 (*sakusha*; 'author')
	著者 (*chosha*; 'writer')
	執筆 (を) (*shippitsu (o)*; 'to write [as a profession]')
	書く (を) (*kaku (o)*; 'to write')
	著す (を) (*arawasu (o)*; 'to write, to publish [a book]')
	more general: 3
	知る (で) (*shiru (de)*; 'to be known for')
	作品 (*sakuhin*; 'work, opus')
	作 (*saku*; 'work of art')
	more specific: 3
	漫画 (*manga*; 'manga')
	小説 (*shousetsu*; 'novel')
	原作者 (*gensakusha*; 'original author')
	related: 4
	発売される (から / より) (*hatsubaisareru (kara/yori)*; 'to be released [for sale]')
	編纂 (を) (*hensan*; 'to compile, edit')
	タイトル (*taitoru*; 'title')
	unrelated: 5
	歴史家 (*rekishika*; 'historian')
	する (を) (*suru (o)*; 'to do')
	される (による) (*sareru (niyoru)*; 'to be done by')
	描く (が) (*egaku (ga)*; 'to paint')
	続く (に) (*tsuzuku*; 'to follow')

`bandMember`	*direct lexicalization*: 1
	バンド (*bando*; 'band')
	more general: 3
	メンバー (*menbaa*; 'member')
	結成する (を/と) (*kesseisuru (o/to)*; 'to form [a team/band etc.]')
	more specific: 10
	ギタリスト (*gitarisuto*; 'guitarist')
	ボーカリスト (*bookarisuto*; 'vocalist')
	ボーカル (*bookaru*; abbreviation of 'vocalist')
	リーダー (*riidaa*; 'leader')
	ヴォーカル (*vuookaru*; alternative form of abbreviation of 'vocalist')
	音楽ユニット (*ongakuyunitto*; "music unit" [special kind of J-Pop band])
	ベーシスト (*beeshisuto*; 'bassist')
	音楽ユニットギタリスト (*ongakuyunittogitarisuto*; 'music unit guitarist')
	ロックバンド (*rokkubando*; 'rock band')
	ユニット (*yunitto*; 'unit')
	related: 3
	脱退する (を) (*dattaisuru (o)*; 'to retire, withdraw from')
	デビューする (として) (*debyuusuru (toshite)*; 'to debut as')
	派生 (*hasei*; 'spin-off')
	unrelated: 3
	what (part of band name/album title)
	人 (*hito*; 'person')
	なる (*naru*; 'to become')
`foundingYear`	*direct lexicalization*: 5
	設立する (に/が/を/は) (*setsuritsusuru (ni/ga/o/wa)*; 'to found')
	創設する (に) (*sousetsusuru (ni)*; 'to found')
	more general: 6
	成立する (に/が) (*seiritsusuru (ni/ga)*; 'to come into existence')
	組織 (に) (*soshiki (ni)*; 'to organize, to construct')
	発足する (に) (*hossokusuru (ni)*; 'start')
	始まる (に/から) (*hajimeru (ni/kara)*; 'to start, to originate')
	more specific: 4
	創業する (に) (*sougyousuru (ni)*; 'to establish [a business]')
	建国する (に/が) (*kenkokusuru (ni/ga)*; 'to found [a nation]')
	独立する (に) (*dokuritsusuru (ni)*; 'to become independent')
	related: 3
	なる (と/に) (*naru (to/ni)*; 'to become')
	分離する (に) (*bunrisuru (ni)*; 'to divide, separate')

	unrelated: 2
	年 (*toshi*; 'year')
	発売 (に) (*hatsubai (ni)*; 'to release [for sale]')
`languageFamily`	*direct lexicalization*: 1
	言語 (*gengo*; 'language')
	more general: 12
	属する (に/の) (*shokusuru (ni/no)*; 'to belong to')
	含む (を) (*fukumu (o)*; 'to contain')
	属す (に) (*shokusu (ni)*; 'to belong to')
	分類する (に) (*bunruisuru (ni)*; 'to classify')
	一つ (*hitotsu*; 'one [of several]')
	一種 (isshu; 'one kind, variety')
	うち (*uchi*; 'inside')
	種 (*kusa*; 'kind, variety')
	ひとつ (*hitotsu*; 'one [of several]')
	中 (*uchi*; 'inside')
	よう (*you*; 'like, such as')
	more specific: 0
	related: 2
	方言 (*hougen*; 'dialect')
	影響 (*eigyou*; 'influence')
	unrelated: 5
	する (を) (*suru (o)*; 'to do')
	現代 (*gendai*; 'contemporary')
	発音 (*hatsuon*; 'pronunciation')
	標準 (*hyoujun*; 'standard')
	ポルトガル (*porutogaru*; 'Portugal')
`locationCity`	*direct lexicalization*: 0
	more general: 5
	置く (に) (*oku (ni)*; 'to put, to place')
	所在する (に) (*shozaisuru (ni)*; 'to be located in')
	存在する (に) (*sonzaisuru (ni)*; 'to exist')
	一つ (*hitotsu*; 'one [of several]')
	拠点 (*kyoten*; 'location, base')
	more specific: 8
	本社 (*honsha*; 'head office')
	設立する (に) (*setsuritsusuru (ni)*; 'to establish, found')
	メーカー (*meekaa*; 'manufacturer')
	会社 (*kaisha*; 'company, corporation')

	本拠地 (*honkyochi*; 'headquarters') 行う (で) (*okonau (de)*; 'to perform, to take place') 創業 (で) (*sougyou (de)*; 'to establish') 構える (に) (*kamaeru (ni)*; 'to set up')
	related: 2
	出身 (*shusshin*; 'person's origin') する (を) (*suru (o)*; 'to do')
	unrelated: 5
	マイケル (*maikeru*; 'Michael [Bloomberg]') 東京都 (*toukyouto*; 'Tokyo metropolitan area') 東京 (*toukyou*; 'Tokyo') 生放送 (*namahousou*; 'live broadcast') the (part of building names)
total	*direct lexicalization*: 12
	more general: 29
	more specific: 25
	related: 14
	unrelated: 20

Table 17: Top 20 entries for the new properties sorted according to how their semantics compare to the meaning of the property at hand. Verbs that occurred several times with different particles are given in one single line, but counted as distinct lemmas.

As figure 23 shows, if the additional verbalizations from table 16 are added to the gold lexicon, precision for the lexicon based on the new properties significantly increases for both filtering strategies. Which number of lemma occurrences or number of entries one should use as one's threshold, i.e. whether one should prefer higher precision or higher recall, again depends on the application area at hand: In case of natural language understanding one would want access to as many different potential verbalizations as possible, hence recall would be more relevant, while in case of natural language understanding one would want to make sure that no incorrect verbalizations are used in the output, which would make precision more important.

Alternative filtering mechanisms, such as those discussed in Walter (2017), may help to further improve precision.

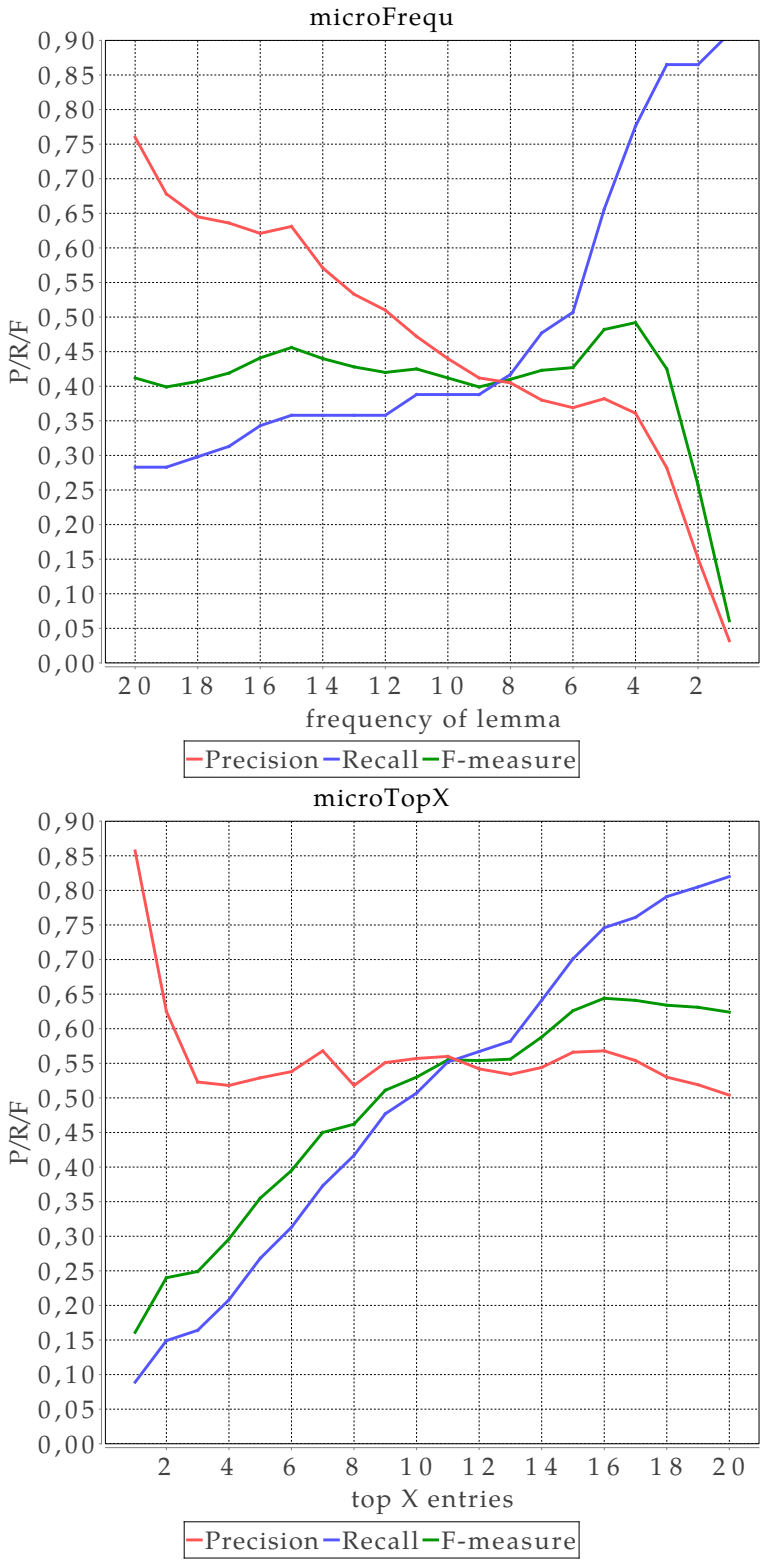


Figure 23: Precision, recall and f-measure for lexicon generated by M-ATOLL for new properties, with additional lemmas from table 16 in gold lexicon

6

DISCUSSION

6.1 COMPARISON OF CROWDSOURCING AND M-ATOLL

As the results from tables 5 and 6 and figure 23 show, with appropriate filtering the lexicons generated by M-ATOLL can reach a higher recall than the crowdsourced lexicon, while precision is better for the latter. Overall, the crowdsourced lexicon seems to be of significantly better quality than the lexicons generated by M-ATOLL, with the former having an optimum f-measure more than ten percentage points higher (0.79) than that of the latter (< 0.65). To a certain degree this is to be expected, since at least when one has access to language-competent workers an approach based on the work of humans will not produce as much noise as a (semi-)automatic approach such as M-ATOLL. Obvious reasons to still use M-ATOLL would be the significantly lower costs and temporal effort, and the fact that M-ATOLL does not depend upon the availability of a crowdsourcing platform with a sufficient amount of workers with adequate language skills. Furthermore, as table 18 shows e.g. for the `yearOfConstruction` property, it seems that crowdsourcing tends to generate comparably few verbalizations whose semantic granularity sticks very closely to that of the English seed verbalizations, while M-ATOLL is able to find a larger variety of verbalizations of differing semantic specificity; as mentioned before, depending on the application area at hand either more general or more specific verbalizations would still be useful. In addition, due to them being close to the original English seed verbalizations, the crowdsourced verbalizations will not necessarily represent the most natural way to express a given property in the target language at hand: For example, for the property `occupation` the English seed lexicon contains the verbalizations 'occupation' and 'to work (as)', and the crowdsourced Japanese verbalizations for that property more or less correspond to these. However, it seems that at least in the Japanese Wikipedia these crowdsourced verbalizations are used only very rarely to express the `occupation` property; instead, during the analysis of the SPARQL queries described in section 5.6.1 it turned out that for the most part `occupation` is not expressed lexically at all, but syntactically through an apposition construction such as the one given in example 34.

(34) 『ふわり』-は、 声優・ 林原めぐみ-の 9枚目-の
fuwari-wa *seiyuu* *hayashibaramegumi-no* *kyuumaime-no*
"Fuwari"-TOP voice.actor Megumi.Hayashibara-POSS ninth-POSS
アルバム。
arubamu
album
"Fuwari" is the ninth album by voice actress Megumi Hayashibara.

Accordingly, M-ATOLL only found very few verbalizations for this property, and those it found were more general than the English seed verbalizations used for crowdsourcing. Hence, M-ATOLL may on the one hand be better at providing more natural sounding verbalizations, and on the other hand be able to indicate when a given property tends to be expressed by other than lexical means.

There are a number of ways crowdsourcing and M-ATOLL could be used together: As already mentioned in chapter 4, M-ATOLL can be used to automatically extract verbalizations and the linguistic information belonging to them from sentences that were retrieved through the crowdsourced translation task. This was our starting point when we adapted M-ATOLL to Japanese, i.e. we first generated SPARQL queries matching the crowdsourced Japanese sentences. Adapting M-ATOLL this way was rather easy, as the syntactic structure of the sentences retrieved through crowdsourcing stuck rather close to the structure of the English seed sentences and showed only little variance. Of course, the SPARQL queries retrieved this way can also in turn be used for generating new entries with M-ATOLL's corpus-based approach. However, due to the structure of the crowdsourced translations being influenced by the English seed sentences, these queries will not necessarily represent the most prototypical way of expressing properties in the target language at hand. For example, as figures 19 and 20 show, the one pattern we defined for verb lexicon entries when adapting M-ATOLL to the crowdsourced translations (`MainVerb`) seems to be not as important overall with respect to this group as e.g. the `RelVerb` pattern. As mentioned before, one problem with our translation-based approach to crowdsourcing ontology lexicons is the rather low variance, and accordingly low recall, of the retrieved verbalizations; in contrast, M-ATOLL is able to find a wider variety of verbalizations, but the resulting lexicon also contains much more noise. Furthermore, in contrast to the crowdsourcing approach, adapting M-ATOLL to another language requires appropriate gold verbalizations already available in the target language, based on which one can define suitable SPARQL queries. Hence, a further possible workflow for retrieving more different verbalizations than through crowdsourcing alone — and for adapting M-ATOLL to a new language at the same

property	*English seed verbalizations used for crowdsourcing*	*matching lemmas found through crowdsourcing*	*matching lemmas found by M-ATOLL*
`occupation`	occupation	活動 (*katsudou*; 'activity')	
	to work (as)	職業 (*shokugyou*; 'occupation')	
		仕事 (*shigoto*; 'work')	
		働く (*hataraku*; 'to work')	
			務める (を) (*tsutomeru (o)*; 'to serve as')
			担当する (は/を) (*tantousuru (wa/o)*; 'to be in charge of')
			作品 (*sakuhin*; 'work, performance')
			知られる (として) (*shirareru (toshite)*; 'to be known as')
			先輩 (*senpai*; 'senior')
`parent`	child	父親 (*chichioya*; 'father [of someone]')	
	father	親 (*oya*; 'parent')	
	daughter	子供 (*kodomo*; 'child')	
	son	母親 (*hahaoya*; 'mother [of someone]')	
	parent	父	父 (*chichi*; 'father')
	mother	母	母 (*haha*; 'mother')
		息子	息子 (*musuko*; 'son')
		子	子 (*kou*; 'child')
		娘	娘 (*musume*; 'daughter')
			長男 (*chounan*; 'eldest son')
			次男 (*jinan*; 'second son')
			三男 (*sannan*; 'third son')
			孫 (*mago*; 'grandson')
`yearOfConstruction`	to construct	建設する (に)	建設する (に) (*kensetsusuru (ni)*; 'to construct')
		建てる (に)	建てる (に) (*tateru (ni)*; 'to build')
			完成する (に) (*kanseisuru (ni)*; 'to complete')
			竣工する (に) (*shunkousuru (ni)*; 'to complete construction')
			開通する (に) (*kaitsuusuru (ni)*; 'to open')
			登場する (に) (*toujousuru (ni)*; 'to enter [a market]')
			作られる (に) (*tsukurareru (ni)*; 'to be produced')
			存在する (に) (*sonzaisuru (ni)*; 'to exist')
			開発 (*kaihatsu*; 'development')
			設立される (に) (*setsuritsusareru (ni)*; 'to be founded')
			投入する (から) (*tounyuusuru (kara)*; 'to introduce')
			建立 (に) (*konryuu (ni)*; 'to build')

Table 18: Matching lemmas found through crowdsourcing and by M-ATOLL (from top 40 generated entries) for three properties

time — would be to first carry out the translation-based crowdsourcing task and then use the retrieved candidate verbalizations for finding SPARQL queries appropriate for the language at hand as described before, i.e. by looking for occurrences of these verbalizations in corpus sentences that contain labels of ontology elements linked by the respective property in a triple store, and check — either manually or semi-automatically — for regularly occurring dependency patterns. These patterns could then be used to find further verbalizations for the properties at hand. Finally, in order to reduce the noise in the resulting lexicon, the additional candidate verbalizations retrieved through M-ATOLL could be used as input to a crowdsourced evaluation task.

7

CONCLUSION AND OUTLOOK

7.1 REQUIREMENTS AND RESEARCH QUESTIONS REVISITED

In this section we look again at the requirements established in the introduction that our approaches to the lexicalization of ontologies should fulfill, and aim to answer the research questions from section 1.3.

7.1.1 *Requirements*

- *The manual effort required by each individual worker involved in the ontology lexicalization process should be as minimal as possible. This concerns both the workers involved in the crowdsourcing approach as well as the pre- and postprocessing stages of both approaches.* After M-ATOLL has been adapted to a new language — in particular after the new language-specific dependency patterns have been specified — no further manual labor is necessary in the lexicon generation process. However, the final lexicon may benefit from manual filtering of the lexicon entries e.g. by a domain expert. With respect to crowdsourcing, given that the candidate verbalizations can be extracted from the translated sentences by means of M-ATOLL, no manual pre- or postprocessing would be necessary. The verbalizations are both generated and evaluated by crowdworkers, through a task design that requires minimal effort for a single task: The workers only have to translate one sentence, or judge the translations of one sentence, respectively, per microtask.

- *The overall costs — i.e. the effort, time and money that need to be spent on ontology lexicalization with the approach at hand — need to be acceptable.* As mentioned before, with M-ATOLL after the initial adaptation to a new language no further manual labor is necessary. Furthermore, applying the system does not come with any monetary costs. With respect to crowdsourcing, the temporal and monetary effort have already been discussed in chapter 4. While our crowdsourcing experiment was more expensive and took longer than generating a Japanese ontology lexicon

with M-ATOLL, it was still less expensive than hiring a professional translator to receive the same results.

- *The entries generated by means of the approach at hand should be of appropriate quality, i.e. they need to contain meaningful verbalizations, plus correct and sufficient linguistic information.* The lexicon generated with M-ATOLL was quite noisy at first, and required further filtering, after which, however, it reached good quality. For our crowdsourcing approach, the precision of the resulting lexicon was very high. In both cases the additional linguistic information is provided by M-ATOLL; the types of information it can extract were described in section 2.5, and include e.g. the part-of-speech, subcategorization frames and the mapping between semantic and syntactic arguments of the candidate verbalization at hand.

- *The approach should be suitable for smaller languages, for which finding enough speakers may be difficult.* For smaller languages the main challenge when adapting M-ATOLL would probably be to find someone with sufficient knowledge of the language at hand to be able to specify the dependency patterns. If these patterns could be generated in a (semi-)automatic way, as described in section 5.6.1, less knowledge about the language at hand would be necessary. With respect to crowdsourcing, from our experiences it seems that it decidedly depends upon the crowdsourcing platform at hand and its demographics whether a sufficient amount of speakers also of smaller languages can be found.

- *The specification format should support multilinguality; in particular, it should be flexible enough to be able to represent linguistic variance e.g. with respect to part-of-speech systems, syntactic constructions etc. among as many different languages as possible.* At least in case of Japanese, lemon has proven to be very suitable for this. For example, due to lemon not being reliant upon a specific linguistic inventory, we could simply use the Japanese part-of-speech tags provided by MeCab in our final lexicons, and did not have to map these e.g. onto English-based part-of-speech tags. Furthermore, it was very easy to define new lexicon entry templates for Japanese, as described in section 5.5, which was also due to lemon not being bound to some fixed inventory of linguistic categories. Hence, lemon is flexible enough to represent at least this language appropriately.

- *In principle it should be possible to specify finer semantic differences among verbalizations that go beyond the ontology elements a given verbalization may refer to. This may be particularly important in cases where the language the ontology identifiers are specified*

in and the lexicon's target language do not match. As has been shown in section 5.6.2, lemon supports this by means of conditions that can be imposed upon the sense of a given candidate verbalization. Furthermore, for verbalizations whose semantics are more specific or more general than that of the property at hand, it would at least in principle be possible to detect this automatically e.g. within a system such as M-ATOLL. However, in case of other mismatches between the semantics of a given candidate verbalization and a given property, detecting these automatically may be rather difficult.

7.1.2 *Research Questions*

1. *Are crowdsourcing and M-ATOLL appropriate for creating ontology lexicons in Non-Indo-European languages of good quality and at acceptable costs?* At least for Japanese, we showed that both M-ATOLL and crowdsourcing are viable approaches to the generation of ontology lexicons. In case of crowdsourcing, however, further experiments would be necessary in order to find an ideal workflow. In addition, it may be worthwhile to further test in the future how crowdsourcing and M-ATOLL can be used together to create ontology lexicons of even better quality.

2. *How do the two approaches compare to each other with respect to their costs and the quality of the resulting lexicons?* This question has already been dealt with in section 6.1: Overall, the quality of the resulting lexicon in terms of precision and f-measure tends to be better in case of crowdsourcing. However, M-ATOLL does not come with additional monetary and temporal costs, and is not dependent upon the availability of a crowdsourcing platform with a sufficient amount of workers proficient in the language at hand. Furthermore, it seems that M-ATOLL is able to generate a higher variance of different verbalizations for a given property than crowdsourcing, for which the retrieved verbalizations stick semantically very closely to the English source elements.

3. *How do the results of M-ATOLL for Japanese compare to the results for English?* This question has been left open in this thesis due to time constraints. However, a comparison between the accuracy measures of the English lexicon shown in Walter (2017, p. 62) and our values for the Japanese lexicon presented in section 5.6.2 shows that the accuracy values for both lexicons are comparable.

4. *Are the external tools and resources required by M-ATOLL available for Non-Indo-European languages? If they are not, or if they differ considerably from corresponding tools and resources for Indo-European languages, how does one cope with that?* We deliberately chose to work with one of the few Non-Indo-European languages for which a comparably large amount of NLP tools and resources are available, hence we did not have to deal with the problem of required resources or tools not being available. For many languages the main problem when wanting to use M-ATOLL with them will probably be the lack of a dependency parser for the respective language; furthermore, basically for all languages other than English a lack of ontology labels available in the respective language may be a problem, even though, as we have discussed in section 5.3.2, such a label sparseness can in principle be overcome. While for Japanese basically all required tools and resources were available, it turned out that the output format of Japanese dependency parsers differs considerably from that of parsers for Indo-European languages. We were able to deal with this problem by modifying the parser output so as to match the input format expected by M-ATOLL's sentence preprocessing module.

5. *Is lemon, the format for the specification of ontology lexicons used by M-ATOLL, flexible enough to represent information on the linguistic properties of Non-Indo-European languages as required by ontology lexicons, even if these differ considerably from those of Indo-European languages?* As already mentioned in the preceding subsection, as far as we can tell from our experiments with Japanese lemon seems to be very well suited for the representation of linguistic information on Non-Indo-European languages.

6. *Is lemon able to represent finer semantic differences among verbalizations that go beyond the ontology element they refer to, such as in cases where there is a degree of mismatch between the exact semantics of the verbalization and the ontology element at hand?* As again already mentioned for the requirements, it seems that lemon is suitable for representing such further aspects of the semantics of a given verbalization, such as when the semantics of the verbalization are more specific or more general than that of the property at hand. The interesting question in this context would be how a system such as M-ATOLL may be able to automatically detect these kinds of semantic mismatches.

7. *Are our handwritten dependency patterns, which are employed by M-ATOLL's corpus-based approach to extract verbalizations, of sufficient quality?* As has been laid out in section 5.6.1, the handwritten patterns seem to be somewhat lacking in terms of the instances of valid verbalizations they are able to match. However, as has also been shown there, this problem could not be really solved through the introduction of further dependency patterns, so one may assume that the handwritten patterns already fairly well represent what is in principle possible in terms of coverage with this pattern-based approach.

8. *Are there alternative, (semi-)automatic approaches to the generation of these patterns with promising results?* Again in section 5.6.1, we described a semi-automatic approach to the generation of dependency patterns. Out of the 14 hand-crafted patterns, four would have also been found this way, while three patterns were newly generated based on this approach. At least four of these seven patterns seem to have a significant positive impact on the quality of the resulting lexicon. One should note that we only looked at the ten most frequently occurring patterns we could retrieve automatically to find dependency patterns to add to our inventory; looking at a larger number of patterns may have allowed us to retrieve a more numerous and comprehensive inventory of dependency patterns semi-automatically.

9. *Which parts of M-ATOLL are language-specific and accordingly need to be adapted to the characteristics of new target languages?* Obviously the most important part of M-ATOLL that needs to be adapted to new languages are the dependency patterns. Apart from that, in case of Japanese we had to add further lexicon entry templates. In addition, while this does not directly concern the M-ATOLL system itself, we needed to adapt the output format of the Japanese dependency parser since it differed from that of the parsers used for Indo-European languages, which is the input format that M-ATOLL's sentence preprocessing module expects.

10. *Is crowdsouring a feasible way of overcoming a lack of available speakers in case of smaller languages?* As already mentioned in the preceding subsection, it decidedly depends upon the crowdsourcing platform at hand and its demographics whether a sufficient amount of speakers of the smaller language under consideration can be found there. At least on CrowdFlower finding enough workers for Japanese, which belongs to the larger languages of the world, took some time, so for smaller languages it may be even more difficult to find sufficient workers.

7.2 FUTURE WORK

With respect to the M-ATOLL system, it may e.g. be worthwhile to try out alternative techniques for finding labels of ontology elements in the given corpus. As mentioned in section 3.3, Marginean and Eniko (2016) in their sentence selection approach employ different matching strategies based on the property at hand — full matching, partial matching and matching based on coreference resolution by an external resource — while at the moment, in M-ATOLL only full matching and in case of English pronoun resolution based on a number of heuristics is available.

One further important future task with respect to M-ATOLL would be to extend the approach to even more languages. While using M-ATOLL with Japanese turned out to be successful, this of course does not mean that in case of other languages no problems could turn up. For example, in case of languages more under-resourced than Japanese problems may arise if for M-ATOLL's corpus-based approach no suitable dependency parser, or even no corpus, would be available, while the label-based approach may be problematic for some languages due to its reliance upon some lexical resource providing synonymy information, which for a lot of languages of the world will not be available.

One weak spot of M-ATOLL are the hand-crafted dependency patterns employed in its corpus-based approach, whose generation is very labor-intensive and time-consuming. Hence, it would be worthwhile to look further into the possibility of generating such dependency patterns (semi-)automatically, as described in section 5.6.1.

So far, we have used M-ATOLL to generate Japanese lexicalizations of object properties and very few select datatype properties. In the future, it would be desirable for M-ATOLL to able to deal with more different types of datatype properties, which in case of languages such as Japanese would e.g. require some kind of mapping between the representations of numeric literals in the triple store and language-specific representations of numbers, such as the Sino-Japanese number system.

With respect to our crowdsourcing experiments, important future work would include dealing with the question whether the alternative workflow presented in section 4.4.2 is really feasible, and how further aspects of its design such as the quality control for the paraphrasing task should be dealt with. In a similar vein, the workflow combining M-ATOLL and crowdsourcing that was described in section 6.1 should be tested. One problem with these further experiments will be to find a suitable crowdsourcing platform with a sufficient amount of Japanese crowdsourcing workers; as mentioned before, CrowdFlower no longer supports limiting one's task to Japanese workers only,

so the best approach would probably be to resort to Japanese crowdsourcing platforms. Both of our approaches — based either on crowdsourcing or M-ATOLL — would definitely profit from a more extensive evaluation. So far, both approaches have only been carried out and tested based on a few properties, and testing both the crowdsourcing approach and the version of M-ATOLL adapted to Japanese with a larger amount of properties may bring more valuable insights into potential problems.

7.3 SUMMARY

In this thesis we explored semi-automatic means of generating ontology lexicons for Non-Indo-European languages. For this purpose we used both crowdsourcing and the M-ATOLL framework to generate Japanese ontology lexicons for excerpts of the DBpedia ontology. First, we presented a two-stage workflow for crowdsourcing ontology lexicons through a translation task, which we tested by generating a Japanese ontology lexicon for ten exemplary ontology elements from DBpedia. A comparison of this lexicon to a manually created one shows that we were able to generate an ontology lexicon of good quality this way, and that in particular for smaller languages, where it may be difficult to find people with sufficient language competency otherwise, generating ontology lexicons this way is a viable option. However, further tests will be necessary to figure out the optimal workflow. Next, we described the adaptations necessary to generate Japanese ontology lexicons with the M-ATOLL framework, which previously had only been used with Indo-European languages. The main language-dependent part of M-ATOLL are the dependency patterns used in its corpus-based approach; apart from that, we also needed to adapt the lexicon entry templates, and we needed to modify the output format of the Japanese dependency parser so as to match the input format expected by M-ATOLL. We used the adapted version of M-ATOLL to generate two small Japanese ontology lexicons for five properties, respectively, which we compared to manually created gold standard lexicons. This comparison showed that when appropriate filtering mechanisms are applied to the generated lexicons, M-ATOLL is able to produce Japanese ontology lexicons of good quality. Comparing the crowdsourced lexicon with that retrieved through M-ATOLL revealed that in terms of precision and f-measure, the crowdsourced lexicon is of better quality. However, the lexicon generated by M-ATOLL provides a better recall, and furthermore, in contrast to crowdsourcing M-ATOLL does not come with additional costs and processing time. Using both approaches conjoined may help bring together the high precision of the crowdsourcing approach with the higher variance of candidate verbalizations retrieved through M-ATOLL.

BIBLIOGRAPHY

Maribel Acosta, Amrapali Zaveri, Elena Simperl, Dimitris Kontokostas, Sören Auer, and Jens Lehmann. Crowdsourcing linked data quality assessment. In *12th International Semantic Web Conference, 21-25 October 2013, Sydney, Australia*, pages 260–276, 2013. URL http://svn.aksw.org/papers/2013/ISWC_Crowdsourcing/public.pdf.

Vamshi Ambati and Stephan Vogel. Can crowds build parallel corpora for machine translation systems? In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, CSLDAMT '10, pages 62–65, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1866696.1866706.

Robert A. Amsler. A taxonomy for english nouns and verbs. In *Proceedings of the 19th Annual Meeting on Association for Computational Linguistics*, ACL '81, pages 133–138, Stroudsburg, PA, USA, 1981. Association for Computational Linguistics. doi: 10.3115/981923.981959. URL http://dx.doi.org/10.3115/981923.981959.

Robert Alfred Amsler. *The Structure of the Merriam-webster Pocket Dictionary*. PhD thesis, 1980. AAI8109129.

Mihael Arcan and Paul Buitelaar. Ontology label translation. In Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff, editors, *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 40–46. The Association for Computational Linguistics, 2013. URL http://aclweb.org/anthology/N/N13/N13-2006.pdf.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *Proceedings of the 6th International The Semantic Web and 2nd Asian Conference on Asian Semantic Web Conference*, ISWC'07/ASWC'07, pages 722–735, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 3-540-76297-3, 978-3-540-76297-3. URL http://dl.acm.org/citation.cfm?id=1785162.1785216.

Núria Bel, Federica Busa, Nicoletta Calzolari, Elisabetta Gola, Alessandro Lenci, Monica Monachini, Antoine Ogonowski, Ivonne Peters, Wim Peters, Nilda Ruimy, Marta Villegas, and Antonio Zampolli. SIMPLE: A general framework for the development of multilingual lexicons. In *Proceedings of the Second International Conference on Language Resources and Evaluation, LREC 2000, 31 May - June 2, 2000, Athens, Greece*. European Language Resources Association, 2000. URL http://www.lrec-conf.org/proceedings/lrec2000/pdf/61.pdf.

Martin Benjamin and Paula Radetzky. Multilingual Lexicography with a Focus on Less-Resourced Languages: Data Mining, Expert Input, Crowdsourcing, and Gamification. In *9th edition of the Language Resources and Evaluation Conference*, 2014.

Bran Boguraev and Ted Briscoe. Large lexicons for natural language processing: Utilising the grammar coding system of ldoce. *Comput. Linguist.*, 13(3-4):203–218, July 1987. ISSN 0891-2017. URL http://dl.acm.org/citation.cfm?id=48160.48162.

Michael R. Brent. Automatic acquisition of subcategorization frames from untagged text. In *Proceedings of the 29th Annual Meeting on Association for Computational Linguistics*, ACL '91, pages 209–214, Stroudsburg, PA, USA, 1991. Association for Computational Linguistics. doi: 10.3115/981344.981371. URL http://dx.doi.org/10.3115/981344.981371.

T. Briscoe, V. de Paiva, and A. Copestake. *Inheritance, Defaults, and the Lexicon*. Cambridge University Press, Cambridge, CA, 1993.

Ted Briscoe and John Carroll. Automatic extraction of subcategorization from corpora. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, ANLC '97, pages 356–363, Stroudsburg, PA, USA, 1997. Association for Computational Linguistics. doi: 10.3115/974557.974609. URL http://dx.doi.org/10.3115/974557.974609.

Chris Callison-Burch and Mark Dredze. Creating speech and language data with amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, CSLDAMT '10, pages 1–12, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1866696.1866697.

John A. Carroll, Guido Minnen, and Ted Briscoe. Can subcategorisation probabilities help a statistical parser? *CoRR*, cmp-lg/9806013, 1998. URL http://arxiv.org/abs/cmp-lg/9806013.

Christian Chiarcos. Olia – ontologies of linguistic annotation. *Semantic Web Journal*, 2012. URL http://semantic-web-journal.net/content/olia-ontologies-linguistic-annotation.

P Cimiano, P Buitelaar, J McCrae, and M Sintek. Lexinfo: A declarative model for the lexicon-ontology interface. *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(1), 2011. ISSN 1570-8268. URL http://www.websemanticsjournal.org/index.php/ps/article/view/182.

Philipp Cimiano, Andreas Hotho, and Steffen Staab. Learning concept hierarchies from text corpora using formal concept analysis. *J. Artif. Int. Res.*, 24(1):305–339, August 2005. ISSN 1076-9757. URL http://dl.acm.org/citation.cfm?id=1622519.1622528.

D.A. Cruse. *Lexical Semantics*. Cambridge University Press, Cambridge, UK, 1986.

Carl G. de Marcken. Parsing the lob corpus. In *Proceedings of the 28th Annual Meeting on Association for Computational Linguistics*, ACL '90, pages 243–251, Stroudsburg, PA, USA, 1990. Association for Computational Linguistics. doi: 10.3115/981823.981854. URL http://dx.doi.org/10.3115/981823.981854.

Gaël Dias and Rumen Moraliyski. Relieving polysemy problem for synonymy detection. In *Proceedings of the 14th Portuguese Conference on Artificial Intelligence: Progress in Artificial Intelligence*, EPIA '09, pages 610–621, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-04685-8. doi: 10.1007/978-3-642-04686-5_50. URL http://dx.doi.org/10.1007/978-3-642-04686-5_50.

M. Ebi and V. Eschbach-Szabo. *Japanische Sprachwissenschaft: Eine Einführung für Japanologen und Linguisten*. Narr Studienbücher. Narr Dr. Gunter, 2015. ISBN 9783823368847. URL https://books.google.de/books?id=50DHoAEACAAJ.

Philip Edmonds and Graeme Hirst. Near-synonymy and lexical choice. *Comput. Linguist.*, 28(2):105–144, June 2002. ISSN 0891-2017. doi: 10.1162/089120102760173625. URL http://dx.doi.org/10.1162/089120102760173625.

Jason M. Eisner. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 1*, COLING '96, pages 340–345, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics. doi: 10.3115/992628.992688. URL http://dx.doi.org/10.3115/992628.992688.

David Faure and Claire Nedellec. Knowledge acquisition of predicate argument structures from technical texts using machine learning: The system asium. In *Proceedings of the 11th European Workshop on Knowledge Acquisition, Modeling and Management*, EKAW '99, pages 329–334, London, UK, UK, 1999. Springer-Verlag. ISBN 3-540-66044-5. URL http://dl.acm.org/citation.cfm?id=645360.650750.

Haim Gaifman. Dependency systems and phrase-structure systems. *Information and Control*, 8(3):304–337, 1965.

Dafydd Gibbon. *Computational Lexicography*. Kluwer Academic Publishers, Dordrecht, 2000.

Roxana Girju, Dan Moldovan, and Adriana Badulescu. Automatic discovery of part-whole relations. *Computational Linguistics*, 32(1):83–135, 3 2006. ISSN 0891-2017.

Gintare Grigonyte. *Building and evaluating domain ontologies: NLP contributions.* PhD thesis, Saarland University, 2010.

Ralph Grishman, Catherine Macleod, and Adam Meyers. Comlex syntax: Building a computational lexicon. In *COLING 1994 Volume 1: The 15th International Conference on Computational Linguistics*, pages 268–272, 1994. URL http://www.aclweb.org/anthology/C94-1042.

Jan Hajič, Martin Čmejrek, Bonnie Dorr, Yuan Ding, Jason Eisner, Dan Gildea, Terry Koo, Kristen Parton, Gerald Penn, Dragomir R. Radev, Owen Rambow, Jan Cuřín, Jiří Havelka, Vladislav Kuboň, and Ivona Kučerová. Final report: Natural language generation in the context of machine translation. 2004.

David G. Hays. Dependency theory: a formalism and some observations. *Language*, 40: 511–525, 1964.

Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th Conference on Computational Linguistics - Volume 2*, COLING '92, pages 539–545, Stroudsburg, PA, USA, 1992. Association for Computational Linguistics. doi: 10.3115/992133.992154. URL http://dx.doi.org/10.3115/992133.992154.

Peter Hellwig. Dependency unification grammar. In *Proceedings of the 11th Coference on Computational Linguistics*, COLING '86, pages 195–198, Stroudsburg, PA, USA, 1986. Association for Computational Linguistics. doi: 10.3115/991365.991423. URL http://dx.doi.org/10.3115/991365.991423.

Hermanus Heringa. *Appositional Constructions*. PhD thesis, 2012.

Graeme Hirst. Ontology and the lexicon. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 269–292. Springer, 2009. ISBN 978-3-540-70999-2. doi: 10.1007/978-3-540-92673-3_12. URL http://dx.doi.org/10.1007/978-3-540-92673-3_12.

Jeff Howe. The rise of crowdsourcing. *Wired Magazine*, 14(6), 06 2006. URL http://www.wired.com/wired/archive/14.06/crowds.html.

Richard Hudson. *English word grammar*. Blackwell, Oxford, UK, 1990.

Richard Hudson. *Language Networks. The new Word Grammar*. Oxford University Press, 2007.

Sean P. Igo and Ellen Riloff. Corpus-based semantic lexicon induction with web-based corroboration. In *Proceedings of the Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics*, UMSLLS '09, pages 18–26, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-34-3. URL http://dl.acm.org/citation.cfm?id=1641968.1641971.

Ann Irvine and Alexandre Klementiev. Using mechanical turk to annotate lexicons for less commonly used languages. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, CSLDAMT '10, pages 108–113, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1866696.1866713.

Marc Kemps-Snijders, Menzo Windhouwer, Peter Wittenburg, and Sue Ellen Wright. Isocat: Corralling data categories in the wild. In *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2008, 26 May - 1 June 2008, Marrakech, Morocco*. European Language Resources Association, 2008. URL http://www.lrec-conf.org/proceedings/lrec2008/summaries/222.html.

Anna Korhonen, Yuval Krymolowski, and Ted Briscoe. A Large Subcategorization Lexicon for Natural Language Processing Applications. In *Proceedings of the 5th international conference on Language Resources and Evaluation*, Genova, Italy, 2006.

Sandra Kübler, Ryan McDonald, Joakim Nivre, and Graeme Hirst. *Dependency Parsing*. Morgan and Claypool Publishers, 2009. ISBN 1598295969, 9781598295962.

Taku Kudo and Yuji Matsumoto. Japanese dependency analysis using cascaded chunking. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002. URL http://www.aclweb.org/anthology/W02-2016.

Bettina Lanser, Christina Unger, and Philipp Cimiano. Crowdsourcing ontology lexicons. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asunción Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*. European Language Resources Association (ELRA), 2016. URL http://www.lrec-conf.org/proceedings/lrec2016/summaries/217.html.

Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation*, SIGDOC '86, pages 24–26, New York, NY, USA, 1986. ACM. ISBN 0-89791-224-1. doi: 10.1145/318723.318728. URL http://doi.acm.org/10.1145/318723.318728.

Marc Light. Corpus processing for lexical acquisition, edited by branimir boguraev and james pustejovsky. *Journal of Logic, Language and Information*, 7(1):111–114, 1998. doi: 10.1023/A:1008203131323. URL http://dx.doi.org/10.1023/A:1008203131323.

Dekang Lin. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th International Conference on Computational Linguistics - Volume 2*, COLING '98, pages 768–774, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics. doi: 10.3115/980432.980696. URL http://dx.doi.org/10.3115/980432.980696.

Dekang Lin, Shaojun Zhao, Lijuan Qin, and Ming Zhou. Identifying synonyms among distributionally similar words. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, pages 1492–1493, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc. URL http://dl.acm.org/citation.cfm?id=1630659.1630908.

D. Lindberg, B. Humphreys, and A. McCray. The unified medical language system. *Methods of Information in Medicine*, 32(4):281–291, 1993. URL http://explorer.csse.uwa.edu.au/reference/browse_paper.php?pid=233282040.

Sergey Lobachev. Top languages in global information production. *Partnership: The Canadian Journal of Library and Information Practice and Research*, 3(2), 2008. ISSN 1911-

9593. doi: 10.21083/partnership.v3i2.826. URL https://journal.lib.uoguelph.ca/index.php/perj/article/view/826.

Catherine Macleod, Adam Meyers, Ralph Grishman, Leslie Barrett, and Ruth Reeves. *Designing a dictionary of derived nominals*. 1997.

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-13360-1.

Anca Nicoleta Marginean and Kando Eniko. Towards lexicalization of dbpedia ontology with unsupervised learning and semantic role labeling. In James H. Davenport, Viorel Negru, Tetsuo Ida, Tudor Jebelean, Dana Petcu, Stephen M. Watt, and Daniela Zaharie, editors, *18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2016, Timisoara, Romania, September 24-27, 2016*, pages 256–263. IEEE Computer Society, 2016. ISBN 978-1-5090-5707-8. doi: 10.1109/SYNASC.2016.048. URL http://dx.doi.org/10.1109/SYNASC.2016.048.

Diana Maynard, Adam Funk, and Wim Peters. Using lexico-syntactic ontology design patterns for ontology creation and population. In *Proceedings of the 2009 International Conference on Ontology Patterns - Volume 516*, WOP'09, pages 39–52, Aachen, Germany, Germany, 2009. CEUR-WS.org. URL http://dl.acm.org/citation.cfm?id=2889761.2889765.

John McCrae, Mauricio Espinoza, Elena Montiel-Ponsoda, Guadalupe Aguado-de Cea, and Philipp Cimiano. Combining statistical and semantic approaches to the translation of ontologies and taxonomies. In *Proceedings of the Fifth Workshop on Syntax, Semantics and Structure in Statistical Translation*, SSST-5, pages 116–125, Stroudsburg, PA, USA, 2011a. Association for Computational Linguistics. ISBN 978-1-932432-99-2. URL http://dl.acm.org/citation.cfm?id=2024261.2024274.

John McCrae, Dennis Spohr, and Philipp Cimiano. Linking lexical resources and ontologies on the semantic web with lemon. Proceedings of the 8th Extended Semantic Web Conference (ESWC), pages 245–259, 2011b. doi: 10.1007/978-3-642-21034-1_17.

Igor Mel'čuk. *Dependency Syntax: Theory and Practice*. State University of New York Press, 1988.

A. Michiels and J. Noël. Approaches to thesaurus production. In *Proceedings of the 9th Conference on Computational Linguistics - Volume 1*, COLING '82, pages 227–

232, Czechoslovakia, 1982. Academia Praha. doi: 10.3115/991813.991849. URL http://dx.doi.org/10.3115/991813.991849.

George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11): 39–41, November 1995. ISSN 0001-0782. doi: 10.1145/219717.219748. URL http://doi.acm.org/10.1145/219717.219748.

Saif M. Mohammad and Peter D. Turney. Crowdsourcing a word-emotion association lexicon. 29(3):436–465, 2013.

Robert Munro and Hal Tily. The start of the art: an introduction to crowdsourcing technologies for language and cognition studies. 2011.

Joakim Nivre. Dependency grammar and dependency parsing. Technical report, Växjö University, 2005. URL http://stp.lingfil.uu.se/~nivre/docs/05133.pdf.

Joakim Nivre, Johan Hall, and Jens Nilsson. Maltparser: A data-driven parser-generator for dependency parsing. In *In Proc. of LREC-2006*, pages 2216–2219, 2006.

Hiroaki Ohshima and Katsumi Tanaka. Real time extraction of related terms by bi-directional lexico-syntactic patterns from the web. In *Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication*, ICUIMC '09, pages 441–449, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-405-8. doi: 10.1145/1516241.1516318. URL http://doi.acm.org/10.1145/1516241.1516318.

Patrick Pantel and Marco Pennacchiotti. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 113–120, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. doi: 10.3115/1220175.1220190. URL http://dx.doi.org/10.3115/1220175.1220190.

Siddharth Patwardhan and Ted Pedersen. Using wordnet-based context vectors to estimate the semantic relatedness of concepts. In *Proceedings of the EACL 2006 Workshop Making Sense of Sense-Bringing Computational Linguistics and Psycholinguistics Together*, volume 1501, pages 1–8, 2006.

Maria Teresa Pazienza and Armando Stellato. An open and scalable framework for enriching ontologies with natural language content. In *Proceedings of the 19th International Conference on Advances in Applied Artificial Intelligence: Industrial, Engineering and*

Other Applications of Applied Intelligent Systems, IEA/AIE'06, pages 990–999, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-35453-0, 978-3-540-35453-6. doi: 10.1007/11779568_106. URL https://doi.org/10.1007/11779568_106.

Andrew Philpot, Eduard Hovy, and Patrick Pantel. The omega ontology. In *In prep*, pages 59–66, 2005.

Judita Preiss, Ted Briscoe, and Anna Korhonen. A system for large-scale acquisition of verbal, nominal and adjectival subcategorization frames from corpora. In John A. Carroll, Antal van den Bosch, and Annie Zaenen, editors, *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*. The Association for Computational Linguistics, 2007. URL http://aclweb.org/anthology-new/P/P07/P07-1115.pdf.

Chu-Ren Huang Laurent Prévot, Nicoletta Calzolari, Aldo Gangemi, Alessandro Lenci, and Alessandro Oltramari. *Ontology and the lexicon: a multi-disciplinary perspective (introduction)*. Studies in Natural Language Processing. Cambridge University Press, April 2010. ISBN 978-0-521-88659-8.

Chris Quirk, Arul Menezes, and Colin Cherry. Dependency treelet translation: Syntactically informed phrasal smt. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 271–279, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: 10.3115/1219840.1219874. URL https://doi.org/10.3115/1219840.1219874.

Marek Rei and Ted Briscoe. Looking for hyponyms in vector space. In Roser Morante and Wen-tau Yih, editors, *Proceedings of the Eighteenth Conference on Computational Natural Language Learning, CoNLL 2014, Baltimore, Maryland, USA, June 26-27, 2014*, pages 68–77. ACL, 2014. ISBN 978-1-941643-02-0. URL http://aclweb.org/anthology/W/W14/W14-1608.pdf.

Stephen D. Richardson, William B. Dolan, and Lucy Vanderwende. Mindnet: Acquiring and structuring semantic information from text. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*, pages 1098–1102, Montreal, Quebec, Canada, August 1998. Association for Computational Linguistics. doi: 10.3115/980691.980749. URL http://www.aclweb.org/anthology/P98-2180.

Mariano Rico and Christina Unger. Lemonade: A web assistant for creating and debugging ontology lexica. In Chris Biemann, Siegfried Handschuh, André Fre-

itas, Farid Meziane, and Elisabeth Métais, editors, *Natural Language Processing and Information Systems - 20th International Conference on Applications of Natural Language to Information Systems, NLDB 2015 Passau, Germany, June 17-19, 2015 Proceedings*, volume 9103 of *Lecture Notes in Computer Science*, pages 448–452. Springer, 2015. ISBN 978-3-319-19580-3. doi: 10.1007/978-3-319-19581-0_45. URL http://dx.doi.org/10.1007/978-3-319-19581-0_45.

Herbert Rubenstein and John B. Goodenough. Contextual correlates of synonymy. *Commun. ACM*, 8(10):627–633, October 1965. ISSN 0001-0782. doi: 10.1145/365628.365657. URL http://doi.acm.org/10.1145/365628.365657.

Hinrich Schütze. Word space. In *Advances in Neural Information Processing Systems 5*, pages 895–902. Morgan Kaufmann, 1993.

Petr Sgall, Eva Hajičová, and Jarmila Panevová. *The Meaning of the Sentence and Its Semantic and Pragmatic Aspects*. Academia/Reidel Publishing Company, Prague, Czech Republic/Dordrecht, Netherlands, 1986.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. Learning syntactic patterns for automatic hypernym discovery. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1297–1304. MIT Press, 2005. URL http://papers.nips.cc/paper/2659-learning-syntactic-patterns-for-automatic-hypernym-discovery.pdf.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 254–263, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1613715.1613751.

Dennis Spohr. *Towards a multifunctional lexical resource: design and implementation of a graph-based lexicon model*. PhD thesis, University of Stuttgart, 2012. URL http://d-nb.info/1017754446.

S. Starosta. *The Case for Lexicase*. Linguistics: Bloomsbury Academic Collections. Bloomsbury Publishing, 2015. ISBN 9781474246712. URL https://books.google.de/books?id=V9DRCgAAQBAJ.

Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual*

Meeting on Association for Computational Linguistics - Volume 1, ACL '03, pages 8–15, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1075096.1075098. URL http://dx.doi.org/10.3115/1075096.1075098.

Lucien Tesnière. *Eléments de Syntaxe Structurale*. Klincksieck, Paris, 1959.

Peter D. Turney. A uniform approach to analogies, synonyms, antonyms, and associations. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 905–912, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. ISBN 978-1-905593-44-6. URL http://dl.acm.org/citation.cfm?id=1599081.1599195.

Christina Unger, John McCrae, Sebastian Walter, Sara Winter, and Philipp Cimiano. A lemon lexicon for dbpedia. Proceedings of 1st International Workshop on NLP and DBpedia, co-located with the 12th International Semantic Web Conference (ISWC 2013), October 21-25, Sydney, Australia. CEUR Workshop Proceedings, 2013. URL http://ceur-ws.org/Vol-1064/.

Evelyne Viegas, Boyan Onyshkevych, Victor Raskin, and Sergei Nirenburg. From submit to submitted via submission: On lexical rules in large-scale lexicon acquisition. In *In Proc. 31st Annual Meeting of the Association for Computational Linguistics*, pages 32–39, 1996.

Sebastian Walter. *Generation of multilingual ontology lexica with M-ATOLL : a corpus-based approach for the induction of ontology lexica*. PhD thesis, 2017.

Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. What is the jeopardy model? a quasi-synchronous grammar for qa. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 22–32, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D07-1003.

Wenbo Wang, Christopher Thomas, Amit Sheth, and Victor Chan. Pattern-based synonym and antonym extraction. In *Proceedings of the 48th Annual Southeast Regional Conference*, ACM SE '10, pages 64:1–64:4, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0064-3. doi: 10.1145/1900008.1900094. URL http://doi.acm.org/10.1145/1900008.1900094.

George Weber. Top languages: The world's 10 most influential languages. *Language Today*, 2, 1997. URL http://www.andaman.org/BOOK/reprints/weber/rep-weber.htm.

Fei Wu and Daniel S. Weld. Open information extraction using wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 118–127, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1858681.1858694.

Yulan Yan, Naoaki Okazaki, Yutaka Matsuo, Zhenglu Yang, and Mitsuru Ishizuka. Unsupervised relation extraction by mining wikipedia texts using information from the web. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 1021–1029, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-46-6. URL http://dl.acm.org/citation.cfm?id=1690219.1690289.

Naoki Yoshinaga and Masaru Kitsuregawa. A self-adaptive classifier for efficient text-stream processing. In Jan Hajic and Junichi Tsujii, editors, *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 1091–1102. ACL, 2014. ISBN 978-1-941643-26-6. URL http://aclweb.org/anthology/C/C14/C14-1103.pdf.

Omar F. Zaidan and Chris Callison-Burch. Crowdsourcing translation: Professional quality from non-professionals. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1220–1229, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-932432-87-9. URL http://dl.acm.org/citation.cfm?id=2002472.2002626.

Zhu Zhang. Mining relational data from text: From strictly supervised to weakly supervised learning. *Inf. Syst.*, 33(3):300–314, May 2008. ISSN 0306-4379. doi: 10.1016/j.is.2007.10.002. URL http://dx.doi.org/10.1016/j.is.2007.10.002.