Adapting the M-ATOLL Methodology for the Generation of Ontology Lexicons to Non-Indo-European Languages: The Case of Japanese

Bettina Lanser, Philipp Cimiano

Semantic Computing Group, CITEC, Bielefeld University, Inspiration 1, 33619 Bielefeld, Germany
E-mail: blanser@cit-ec.uni-bielefeld.de, cimiano@cit-ec.uni-bielefeld.de

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 paper we present an alternative approach to generating ontology lexicons by means of the framework M-ATOLL. So far, M-ATOLL has been used with Indo-European languages that share a large set of common characteristics. We explore if M-ATOLL can also be used fruitfully with Non-Indo-European languages; for this purpose, we use M-ATOLL to generate a Japanese ontology lexicon for DBpedia.

Keywords: Ontology lexicalization; M-ATOLL; DBpedia

1. Introduction

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. Of course, in order to e.g. map the natural language input onto the elements of the conceptual knowledge repository at hand, 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 extent, 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.

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. 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 possible 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. Furthermore,

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

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 (McCrae et al., 2011a; Arcan & Buitelaar, 2013), corresponding systems have not yet reached an accuracy sufficient to produce high-quality lexicons off the shelf.

Therefore, this paper will deal with an alternative approach to ontology lexicalization that requires less manual effort: We will look at M-ATOLL (Multilingual, Automatic inducTion of OntoLogy Lexica; Walter, 2017), a framework for the (semi-)automatic generation of ontology lexicons in the RDF-based lemon format (McCrae et al., 2011b). Another main topic of this paper 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 investigated 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. Finally, we investigated 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 used M-ATOLL to generate a Japanese ontology lexicon for excerpts from DBpedia's ontology. 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 paper we wanted to focus on problems with language portation that are more directly related to the structure of M-ATOLL and lemon, respectively.

2. M-ATOLL

M-ATOLL (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 different approaches.

The corpus-based approach, which is M-ATOLL's main approach, lexicalizes only properties and is based on a dependency-parsed text corpus whose sentences M-ATOLL tries to match to predefined, language-specific dependency 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 .
```

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 SPARQL queries. 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.

So far, all 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, this paper will deal with adapting M-ATOLL's corpus-based approach to Japanese.

3. Adapting M-ATOLL to Japanese

3.1 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.[2] We extract the sentences from the XML file with an already existing script.[3] We then run the morphological analyzer MeCab[4] on the 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;[5] its output for the following example is shown in Figure 1.

(1) *1943*年、 ロスアラモス国立研究所-を 建設した。
1943.year Los.Alamos.National.Laboratory-DOBJ constructed
In 1943, [someone] constructed the Los Alamos National Laboratory.

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 1 multi-word unit 0, which contains

[2] https://dumps.wikimedia.org/jawiki/
[3] http://medialab.di.unipi.it/wiki/Wikipedia_Extractor
[4] http://taku910.github.io/mecab/
[5] 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 1: Output of dependency parser J.DepP for example sentence 1

the tokens 1943, 年 and a comma, depends upon multi-word unit 2, which consists of the tokens 建設, し, た 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 1 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.

part-of-speech	part-of-speech subcategory 1	inflection type
名詞 noun	サ変接続 verbal (nouns that can form verbs by being followed by する or related verbs)	
動詞 verb	自立 main (i.e. non-auxiliary)	特殊・デス copula verb です 特殊・ダ copula verb だ
助詞 particle	係助詞 dependency (comprises topic marker は) 連体化 adnominalizer (non-possessive の that joins nouns together)	

Table 1: Part-of-speech and inflection type tags used in the SPARQL queries

One of the tasks of M-ATOLL's sentence preprocessing component is to turn the input sentences into RDF. Since the Malt parser,[6] which had been used for dependency parsing the English and Spanish input to M-ATOLL (Walter, 2017: p. 144), uses the CoNLL format[7] as its output format, the sentence preprocessing component was already able

[6] http://www.maltparser.org/userguide.html

[7] http://ilk.uvt.nl/conll/

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 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 1 is again shown in Figure 2, this time in the CoNLL format. Table 2 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 3 shows how the bunsetsu-based dependency structure of sentence 1 would be transformed into a token-based structure. The remaining columns of the CoNLL format were left empty: 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,[8] which is a revised version of CoNLL aimed at being able to represent a larger variety of different languages:[9] 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 2. The remaining features are not specified for multi-word units.

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

[9] 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 2: 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 2: Output of dependency parser J.DepP for example sentence 1, turned into CoNLL format

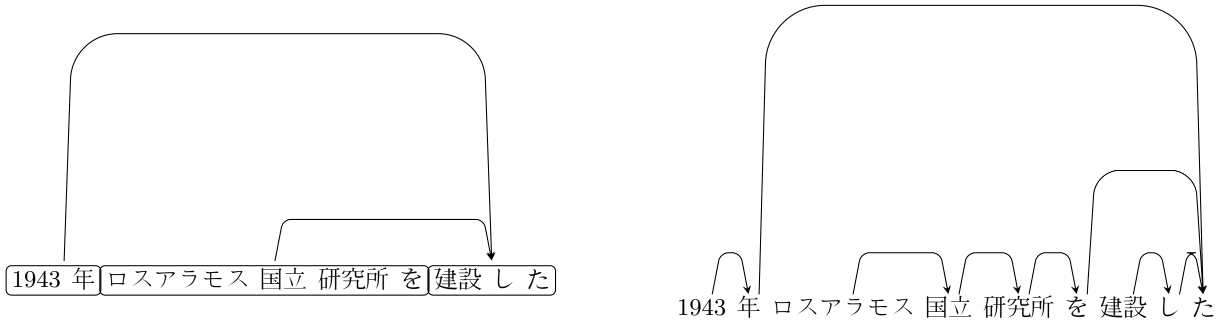

1943 年 ロスアラモス 国立 研究所 を 建設 し た
1943 年 ロスアラモス 国立 研究所 を 建設 し た

Figure 3: Exemplary transformation of bunsetsu-based into token-based dependency structure. The boxes indicate bunsetsu boundaries.

3.2 Dependency Patterns

3.2.1 Overview

We first manually defined eleven dependency patterns for Japanese in terms of SPARQL queries. 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.

The patterns were retrieved based on five example properties from DBpedia's ontology, `parent`, `occupation`, `yearOfConstruction`, `crosses` and `nationality`. The approach to generating the patterns was similar to that described in Walter (2017: 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 manually compiled a set of gold verbalizations our SPARQL patterns should be able to find. In part, we used verbalizations found through crowdsourcing as described in Lanser et al. (2016) for this.
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:

(2) ヘンリー2世-の 母親 である 皇后 マティルダ-は これ-に
Henry.II-POSS mother COP empress Mathilda-TOP this-IOBJ
反対した
opposed

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

(3) 宮崎吾朗-の 父親 である 宮崎駿-は 『ゲド戦記』-の
Goro.Miyazaki-POSS father COP Hayao.Miyazaki-TOP "Earthsea"-POSS
古く-から-の ファン であり
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.

(4) 木曽川-の 橋 である 愛岐大橋-は 慢性的な 渋滞-が
Kiso.river-POSS bridge COP Aichi.Bridge-TOP frequent congestion-SUBJ
発生している
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.

3.2.2 Noun Patterns

Similarly to that which has been described for English, German and Spanish in Walter (2017: 55), most patterns we were able to identify for noun lemmas correspond either to an appositive or a copula construction. With respect to copula constructions we defined two SPARQL queries corresponding in English to the constructions *[e1] is the [lemma] of [e2]* (e.g. *Lydia Hearst is the child of Patty Hearst*) and *The [lemma] of [e2] is [e1]* (e.g. ボブ・ショーの職業はジャーナリスト *The occupation of Bob Shaw [is] journalist*). 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 (*[e1]'s [lemma][e2]* and *[e1] is a NN of [e2][lemma]*) 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. We only found one further pattern that did not belong to either of these two groups, 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.

3.2.3 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, and one pattern for verbs in relative clauses, respectively.

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 5 and 6 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 or the topic particle は 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 7 to 10 below: It turned out that such clauses regularly contained gold lemmas (7) or expressed the desired relation between the entities from the triple store by some other matching verbalization (9).

(5) *property:* `parent`

安楽公主-と共に	中宗-を	毒殺した
Yasushi.Kura-along.with	Nakasune-DOBJ	poisoned

[someone] poisoned Nakasune along with Yasushi.Kura

(6) *property:* `occupation`

会長職-を	李健熙-に	返上した
chairman.position-DOBJ	Lee.Kun.he-IOBJ	gave.up

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

(7) *property:* `yearOfConstruction`

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

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

(8)

グラニット鉄道-を	1826年4月1日-に	着工した
Granite.railway-DOBJ	1826.year.4.month.1.day-on	started

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

(9) *property:* *`occupation`*

声優-として	植田佳奈-が	採用された
voice.actor-as	Kana.Ueda-SUBJ	was.employed

Kana Ueda was employed as a voice actor

(10)

声優-として	植田佳奈-を	採用した
voice.actor-as	Kana.Ueda-DOBJ	employed

[someone] employed Kana Ueda as a voice actor

3.3 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. 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
- part-of-speech: common noun
- subcategorization frame: noun AdP frame
  arguments:

    - copulative argument e_1
    - adpositional object e_2 with adposition の
- 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 7 the `PassiveVerb` template would be invoked and the following entry would be created:

- canonical form: 採用される
- part-of-speech: verb
- subcategorization frame: passive AdP frame
  arguments:
    - subject e_1
    - adpositional object e_2 with adposition として
- 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, 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 7 or 9 this would lead to entries without a subject.

4. Evaluation

4.1 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 3, together with counts of how often each gold lemma was 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 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 (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 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.

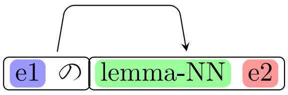
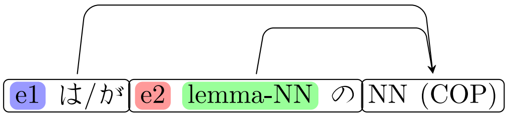
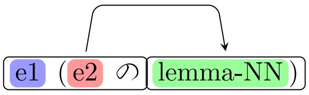
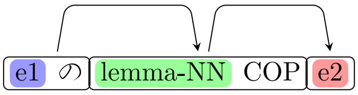
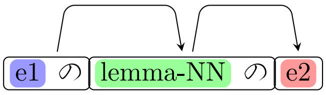
For each instance of a gold lemma found between or behind a triple subject and object, we constructed the minimal path between these three elements within the sentence's dependency tree, and grouped together all instances that share the same path structure.

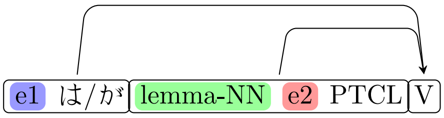
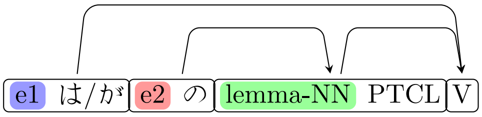
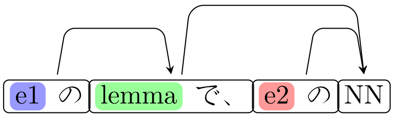
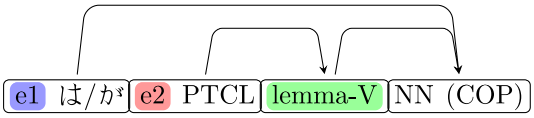
This analysis showed that the vast majority of dependency patterns occurs only once. Basing new SPARQL queries or modifications to existing queries on patterns that only occur a few times overall would probably not be very worthwhile.

Therefore, we looked at the 10 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 4.

property	# sent.s	verbalization	# of instances of lemma between/after triple subject and object	# sent.s found through SPARQL queries	% coverage
`crosses`	241	跨ぐ (to step over, to bridge)	5	2	40
		架かる (to span, to cross)	91	12	13.19
		かかる (writing variant of 架かる)	17	2	11.76
		またがる (to extend over)	1	1	100
		渡る (to cross over)	20	2	10
`nationality`	4660	国籍 (nationality)	9	2	22.22
		出身 (person’s origin)	283	38	13.43
		生まれ (birthplace)	33	3	9.09
`occupation`	9531	仕事 (work)	58	0	0
		職業 (occupation)	10	0	0
		職 (job, position)	16	0	0
		生業 (job)	0	0	-
		勤める to work (for)	9	1	11.11
		務める to serve (as)	342	16	4.68
		活動 (activity)	311	5	1.61
		働く (to work)	9	0	0
`yearOfConstruction`	281	完成 (completion)	11	4	36.36
		竣工 (completion of construction)	2	0	0
		建設 (construction)	16	3	18.75
		建てる (to build)	5	3	60
`parent`	11,831	子供 (child)	104	2	1.92
		子 (child [of someone])	948	31	3.27
		父親 (father [of someone])	54	3	5.56
		父 (father)	651	69	10.60
		娘 (daughter)	928	65	7.00
		息子 (son)	1457	179	12.29
		親 (parent)	9	0	0
		母 (mother)	446	17	3.81
		母親 (mother [of someone])	49	1	2.04

Table 3: 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

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

	ウァレリアヌス-は 息子 Valerian-TOP son ガッリエヌス-を Galienus-DOBJ ローマ帝国-の Roman.Empire-POSS 西半分-を 任せた 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 Rajaram-TOP シヴァージー-の Shivay-POSS 息子-として 生まれた son-as 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 Romanos.II-POSS daughter で、 バシレイオス2世-の COP Basilius.II-POSS 妹。 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、 Change.bridge-TOP セーヌ川-に 架かる Seine-IOBJ to.cross 橋 である。 bridge COP Pont au Change is a bridge that crosses the Seine.	38	no (new pattern created)

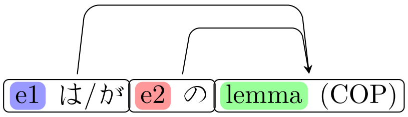
	クレオメネス3世-は Cleomenes.III-TOP レオニダス2世-の 息子 Leonidas.II-POSS son である。 COP Cleomenes III is the son of Leonidas II.	37	yes

Table 4: Most frequently occurring patterns over all lemmas from gold lexicon

Four out of these 10 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 10 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 5 in the Appendix shows how these modifications and the introduction of the three new SPARQL queries influence the number of lemma instances found by M-ATOLL.

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 previously found frequently 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. 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 4 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.

4.2 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 4, while the results for the five new example properties are depicted in Figure 5. As mentioned before, the entries created by M-ATOLL should be filtered in some way; we looked at a filtering strategy filtering strategies, both based on the number of times a given lemma has been found in the corpus: 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.

So for example, 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. In contrast to other filtering strategies where e.g. only lemmas are included in the final lexicon that occur a certain number of times, this 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 Figure 4 and 5, 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 4.1.

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 6.

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 a mismatch in semantic granularity: Some of these verbalizations are more specific than the property at hand, such as ボーカリス

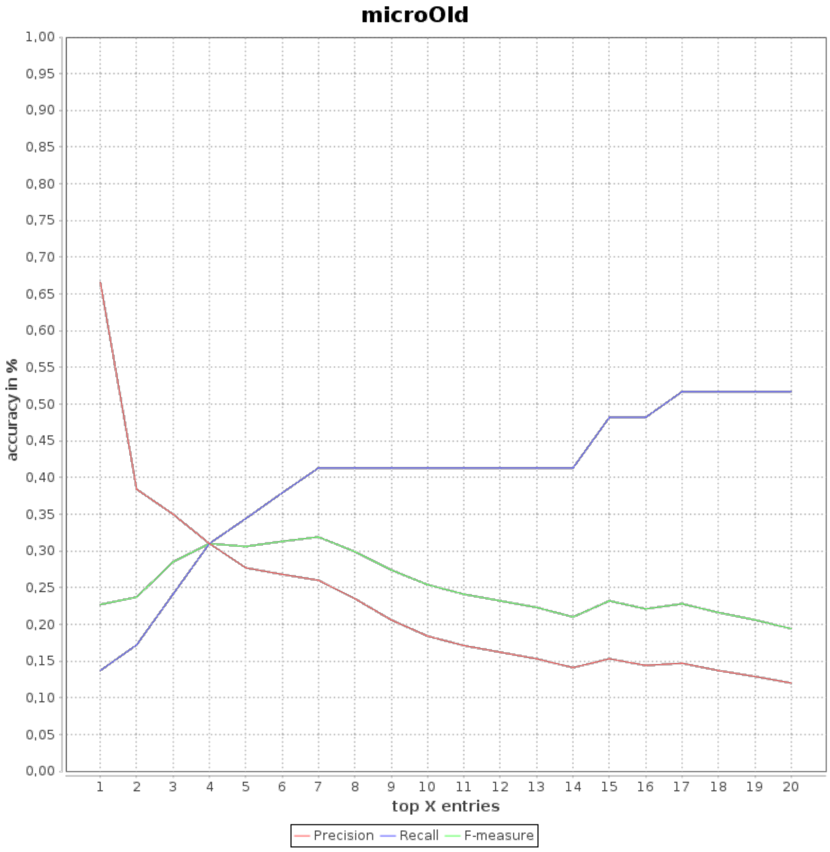


Figure 4: Precision, recall and f-measure for the lexicons generated by M-ATOLL for the old example properties; entries are filtered based on where in a list sorted according to the number of lemma occurrences in which they occur

ト (*vocalist*) as a verbalization of `bandMember`, while in other cases they are considerably more general, such as 一つ (*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 (*one [of several]*) in スペイン語はロマンス諸語の一つです *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 (*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.

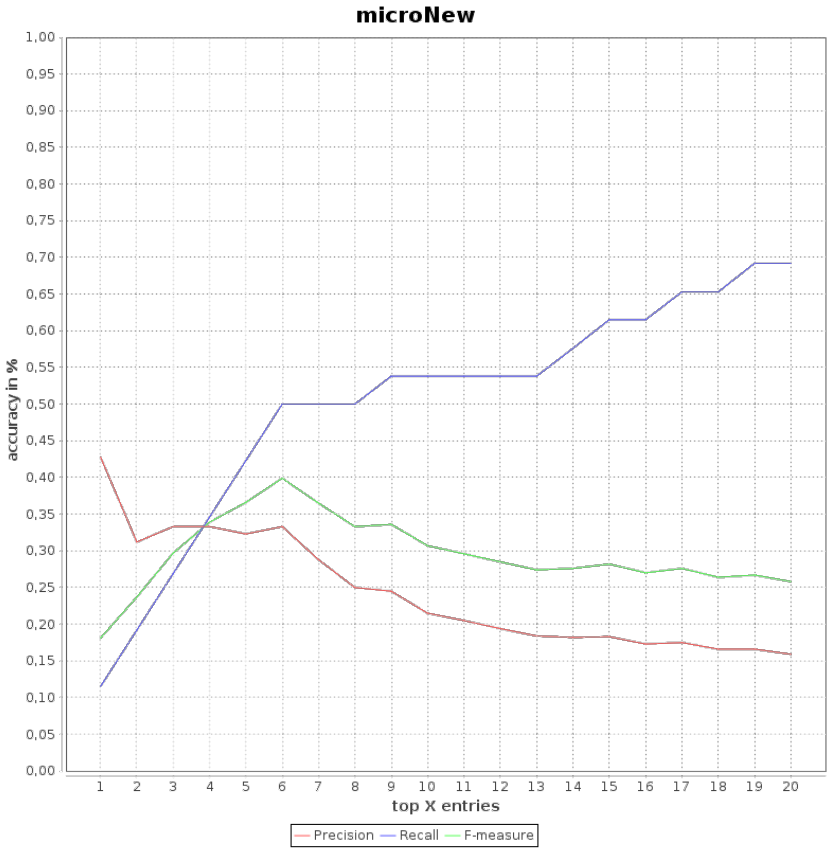


Figure 5: Precision, recall and f-measure for the lexicons generated by M-ATOLL for five new example properties; entries are filtered based on where in a list sorted according to the number of lemma occurrences in which they occur

As Figure 6 shows, if the additional verbalizations from table 6 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.

5. Conclusion

In this paper we explored how M-ATOLL can be used to generate ontology lexicons for Non-Indo-European languages. For this purpose we used M-ATOLL to generate a Japanese ontology lexicon for DBpedia. The three main aspects that required manual work were the adaptation of the output format of the Japanese dependency parser to the input format expected by M-ATOLL, the generation of the language-specific dependency

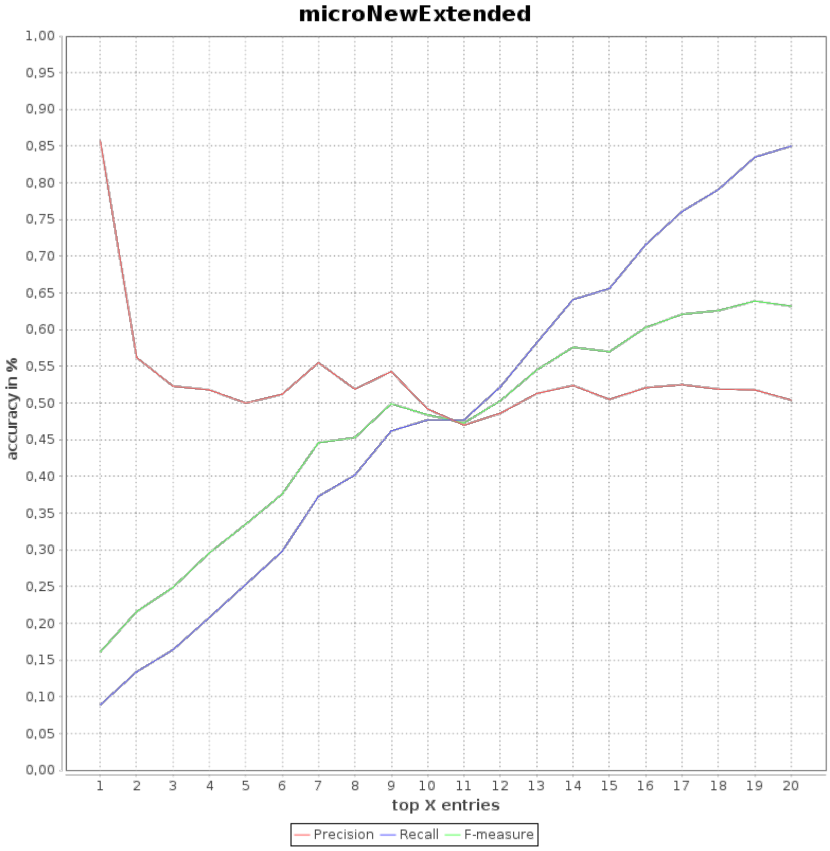


Figure 6: Precision, recall and f-measure for lexicon generated by M-ATOLL for new properties, with additional lemmas from Table 6 in gold lexicon

patterns required by M-ATOLL's corpus-based approach, and the specification of new lexicon entry templates. We showed how the most laborious of these three tasks, the generation of dependency patterns, can be partly automatized in order to reduce the temporal effort. We could show that M-ATOLL is a viable approach to the generation of ontology lexicons also for Non-Indo-European languages. Furthermore, due to it not being reliant on some specific grammatical framework or inventory of linguistic categories, lemon, the format lexicons generated by M-ATOLL are specified in, turned out to be very suitable for being used with Japanese, as can be seen e.g. from the ease with which we could generate new lexicon entry templates.

6. Acknowledgements

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).

7. References

Arcan, M. & Buitelaar, P. (2013). Ontology Label Translation. In L. Vanderwende, H.D. III & K. Kirchhoff (eds.) *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*. The Association for Computational Linguistics, pp. 40–46. URL http://aclweb.org/anthology/N/N13/N13-2006.pdf.

Cimiano, P., Buitelaar, P., McCrae, J. & Sintek, M. (2011). LexInfo: A Declarative Model for the Lexicon-Ontology Interface. *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(1). URL http://www.websemanticsjournal.org/index.php/ps/article/view/182.

Heringa, H. (2012). *Appositional Constructions*. Ph.D. thesis, Rijksuniversiteit Groningen.

Lanser, B., Unger, C. & Cimiano, P. (2016). Crowdsourcing Ontology Lexicons. In N. Calzolari, K. Choukri, T. Declerck, S. Goggi, M. Grobelnik, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk & S. Piperidis (eds.) *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*. European Language Resources Association (ELRA), pp. 3477–3484. URL http://www.lrec-conf.org/proceedings/lrec2016/summaries/217.html.

McCrae, J., Espinoza, M., Montiel-Ponsoda, E., Aguado-de Cea, G. & Cimiano, P. (2011a). 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. Stroudsburg, PA, USA: Association for Computational Linguistics, pp. 116–125. URL http://dl.acm.org/citation.cfm?id=2024261.2024274.

McCrae, J., Spohr, D. & Cimiano, P. (2011b). Linking Lexical Resources and Ontologies on the Semantic Web with lemon. In *Proceedings of the 8th Extended Semantic Web Conference (ESWC)*. pp. 245–259.

Miller, G.A. (1995). WordNet: A Lexical Database for English. *Commun. ACM*, 38(11), pp. 39–41. URL http://doi.acm.org/10.1145/219717.219748.

Prévot, C.R.H.L., Calzolari, N., Gangemi, A., Lenci, A. & Oltramari, A. (2010). *Ontology and the lexicon: a multi-disciplinary perspective (introduction)*. Studies in Natural Language Processing. Cambridge University Press, pp. 3–28.

Walter, S. (2017). *Generation of multilingual ontology lexica with M-ATOLL : a corpus-based approach for the induction of ontology lexica*. Ph.D. thesis, Universität Bielefeld.

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 (to step over, to bridge)	5	2	2	40	40
	架かる (to span, to cross)	91	12	47	13.19	51.65
	かかる (writing variant of 架かる)	17	2	4	11.76	23.53
	またがる (to extend over)	1	1	1	100	100
	渡る (to cross over)	20	2	3	10	15
nationality	国籍 (nationality)	9	2	2	22.22	22.22
	出身 (person's origin)	283	38	119	13.43	42.05
	生まれ (birthplace)	33	3	6	9.09	18.18
occupation	仕事 (work)	58	0	0	0	0
	職業 (occupation)	10	0	0	0	0
	職 (job, position)	16	0	0	0	0
	生業 (job)	0	0	0	-	-
	勤める to work (for)	9	1	1	11.11	11.11
	務める to serve (as)	342	16	40	4.68	11.70
	活動 (activity)	311	5	6	1.61	1.93
	働く (to work)	9	0	0	0	0
yearOfConstruction	完成 (completion)	11	4	5	36.36	45.45
	竣工 (completion of construction)	2	0	1	0	50
	建設 (construction)	16	3	3	18.75	18.75
	建てる (to build)	5	3	3	60	60
parent	子供 (child)	104	2	36	1.92	34.62
	子 (child [of someone])	948	31	102	3.27	10.76
	父親 (father [of someone])	54	3	8	5.56	14.81
	父 (father)	651	69	106	10.60	16.28
	娘 (daughter)	928	65	102	7.00	10.99
	息子 (son)	1457	179	337	12.29	23.13
	親 (parent)	9	0	0	0	0
	母 (mother)	446	17	24	3.81	5.38
	母親 (mother [of someone])	49	1	2	2.04	4.08

Table 5: 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

property	*original lemmas*	*additional lemmas found in top 20 entries of lexicon generated by M-ATOLL*
`author`	著者 (writer)	著す (to write [a book])
	書く (to write)	漫画 (manga)
	作家 (novelist)	小説 (novel)
	著作家 (author)	執筆 (writing [as a profession])
	作品 (work, opus)	原作者 (original author)
	作 (work [of art])	
	作者 (author)	
`bandMember`	バンド メンバー (band member)	ギタリスト (guitarist)
	メンバー (member)	ボーカリスト (vocalist)
	所属 (to belong to [used for humans])	ボーカル (abbrev. of vocalist)
		リーダー (leader)
		結成 (to form [a group of people, e.g. band, team])
		ヴォーカル (altern. writing form of abbrev. of vocalist)
		音楽ユニット ("music unit"; certain type of J-Pop band)
		ベーシスト (bassist)
		ユニット ("unit")
		音楽ユニットギタリスト (music unit guitarist)
		ロックバンド (rock band)
`foundingYear`	設立 (founding)	組織 (organization, construction)
	創立 (establishment)	発足 (start)
	創設 (founding)	建国 (founding of a nation)
	創始 (creation)	創業 (establishment [of a business])
	成立 (coming into existence)	独立 (becoming independent)
		始まる (to start)
`languageFamily`	属す (to belong to)	一種 (one kind, variety)
	言語 (language)	一つ (one [of several])
	含む (to include)	分類 (classification)
		ひとつ (*writing variant of* one)
		種 (kind, variety)
`locationCity`	所在する (to be located)	置く (to put, to place)
	都市 (city)	本社 (head office)
	場所 (location)	存在する (to exist)
		設立 (founding)
		会社 (company, corporation)
		本拠地 (headquarters)
		行う (to perform, to take place)
		創業 (establishment [of a business])
		構える (to set up)
		一つ (one [of several])

Table 6: Entries of gold lexicon for new properties