International Journal on Artificial Intelligence Tools
Vol. 24, No. 2 (2015) 1540010 (36 pages)

DOI: 10.1142/S0218213015400102

World Scientific
www.worldscientific.com

Automatic Extraction of Semantic Relations from Wikipedia

Patrick Arnold* and Erhard Rahm†
Department of Computer Science, Leipzig University, Augustusplatz 10
Leipzig, 04109, Germany
**arnold@informatik.uni-leipzig.de*
†*ahm@informatik.uni-leipzig.de*

Received 12 September 2014
Accepted 22 December 2014
Published 13 April 2015

We introduce a novel approach to extract semantic relations (e.g., is-a and part-of relations) from Wikipedia articles. These relations are used to build up a large and up-to-date thesaurus providing background knowledge for tasks such as determining semantic ontology mappings. Our automatic approach uses a comprehensive set of semantic patterns, finite state machines and NLP techniques to extract millions of relations between concepts. An evaluation for different domains shows the high quality and effectiveness of the proposed approach. We also illustrate the value of the newly found relations for improving existing ontology mappings.

Keywords: Information extraction; semantic relations; natural language processing; background knowledge; thesaurus; Wikipedia.

1. Introduction

Background knowledge plays an important part in information integration, especially in ontology matching and mapping, aiming at finding semantic correspondences between concepts of related ontologies. There are numerous tools and approaches for matching ontologies that mostly focus on finding pairs of semantically equivalent concepts.[29,5,28,9] Most approaches apply a combination of techniques to determine the lexical and structural similarity of ontology concepts or to consider the similarity of associated instance data. The lexical or string similarity of concept names is usually the most important criterion. Unfortunately, in many cases the lexical similarity of concept names does not correlate with the semantic concept similarity due to uncoordinated ontology development and the high complexity of language. For example, the concept pair (*car*, *automobile*) is semantically matching but has no lexical similarity, while there is the opposite situation for the pair (*table*, *stable*). Hence, background knowledge sources such as synonym tables, thesauri and dictionaries are frequently used and vital for ontology matching.

The dependency on background knowledge is even higher for *semantic ontology matching* where the goal is to identify not only pairs of equivalent ontology concepts, but all related concepts together with their semantic relation type, such as is-a or part-of. Determining semantic relations obviously results in more expressive mappings that are an important prerequisite for advanced mapping tasks such as ontology merging[30,31] or to deal with ontology evolution.[19,15] Table 1 lists the main kinds of semantic relations together with examples and the corresponding linguistic constructs. The sample concept names show no lexical similarity so that identifying the semantic relation type has to rely on background knowledge such as thesauri.

Table 1. Semantic concept relations.

Relation Type	Example	Linguistic Relation
equal	river, stream	Synonyms
is-a	car, vehicle	Hyponyms
has-a	body, leg	Holonyms
part-of	roof, building	Meronyms

Relatively few tools are able to determine semantic ontology mappings, e.g., S-Match,[14] TaxoMap,[18] ASMOV[22] and AROMA,[8] as well as our own approach.[2] All these tools depend on background knowledge and currently use WordNet as the main resource. Our approach[2] uses a conventional match result and determines the semantic relation type of correspondences in a separate enrichment step. We determine the semantic relation type with the help of linguistic strategies (e.g., for compounds such as "personal computer" is-a "computer") as well as background knowledge from the repositories WordNet (English language), OpenThesaurus (German language) and parts of the UMLS (medical domain). Together with the match tool COMA[23] for determining the initial mapping, we could achieve mostly good results in determining the semantic relation type of correspondences. Still, in some mapping scenarios recall was limited since the available repositories, including WordNet, did not cover the respective concepts. Based on the previous evaluation results, we see a strong need to complement existing thesauri and dictionaries by more comprehensive repositories for concepts of different domains with their semantic relations.

To build up such a repository automatically, we aim at extracting semantic correspondences from Wikipedia which is the most comprehensive and up-to-date knowledge resource today. It contains almost any common noun of the English language, and thus presumably most concept names. Articles are user-generated and thus of very good quality in general. Furthermore, Wikipedia content can be accessed free of charge.

The rationale behind our approach is based on the observation that definitions in dictionaries or encyclopedias have quite a regular structure. In its classic form, a concept C is defined by a hypernym C', together with some attributes describing

the differences between C and C'. As an example, consider the following Wikipedia definition of *bicycle*:

> *A bicycle, often called a bike, is a human-powered, pedal-driven, single-track vehicle, having two wheels attached to a frame, one behind the other.*

This definition provides (a) the hypernym of *bike*, which is a *vehicle*, and (b) several attributes to distinguish a bike from the more general concept *vehicle*. While some attributes like *human-powered* or *pedal-driven* are not relevant for ontology mapping, some attributes express part-of relations that are indeed valuable. The phrase *having two wheels attached to a frame*, for instance, expresses that a bike has wheels and a frame (wheels part-of bike, frame part-of bike). Therefore, definition sentences can provide both is-a and part-of (or its complementary type has-a) relations. Additionally, the definition above provides a synonym relation, as the terms *bicycle* and *bike* are obviously equivalent because of the expression "*often called*". From a single definition, we can thus extract three relations of different types: equal, is-a, part-of/has-a.

In our work we will show how we can discover the mentioned relations in Wikipedia definition sentence and how we extract the words that take part in such a relation, e.g. {bike, bicycle} is-a {single-track vehicle}. In particular, we make the following contributions:

- We present a novel approach to extract semantic concept correspondences from Wikipedia articles. We propose the use of finite state machines (FSM) to parse Wikipedia definitions and extract the relevant concepts.
- We use a comprehensive set of *semantic patterns* to identify all kinds of semantic relations listed in Table 1. The proposed approach is highly flexible and extensible. It can also extract multiple relations from a single Wikipedia article.
- We show how we can distinguish between entitiy articles and concept articles by using the categories in which articles are listed.
- We evaluate our approach against different subsets of Wikipedia covering different domains. The results show the high effectiveness of the proposed approach to determine semantic concept relations.
- We provide a theoretic evaluation on an existing mapping, showing new correspondences that can be resolved by the knowledge gathered from Wikipedia.

In the next section we discuss related work. Section 3 introduces the notion of semantic patterns and outlines which kinds of patterns we use for discovering semantic relations. Section 4 describes the new approach to extract semantic relations from Wikipedia in detail. In Section 5 we evaluate the approach for different test cases from different domains. Finally, we briefly report on applying our approach to the entire Wikipedia and on the use of the new relations for improving existing ontology mappings (Section 6) before we conclude with a summary and outlook (Section 7).

2. Related Work

Overcoming the large gap between the formal representation of real-world objects (resp. concepts) and their actual meaning is still an open problem in computer science. Lexicographic strategies, structured-based strategies and instance data analysis were successfully implemented in various matching tools, but in many mapping scenarios these strategies do not suffice and state-of-the-art tools can neither determine a complete mapping, nor can they prevent false correspondences. For this reason, background knowledge sources are highly important, as they can improve the mapping quality where generic strategies reach their limits. Hence, a large amount of research has been dedicated to making background knowledge available in diverse resources. Aleksovski *et al.* analyzed the value of background knowledge for ontology mapping in detail.[1] In particular, they showed that a background ontology can significantly improve match quality for mapping rather flat taxonomies without much lexicographic overlap.

The previous approaches for determining background knowledge and the resulting background resources can broadly be classified according to the following criteria:

- **Development:** Manual vs. (semi-) automatic
- **Area:** General vs. domain-specific language
- **Data:** Concept data vs. instance/entity data
- **Number of Languages:** Monolingual vs. multilingual
- **Size/Extent:** Smaller (incomplete) vs. larger (near-complete)
- **Availability:** Free vs. commercial.

In addition to these criteria, there are further differentiating aspects such as the reliability of the provided information or the kind of relationships between concepts or entities (simple links vs. semantic relations such as *equal*, *is-a*, *part-of*, *related*). Some features can be further divided, e.g., manually generated resources can be created by experts or collaboratively by a community of laymen. Also, some features are interrelated, e.g., a semi-automatically generated resource may be of larger size than a manually created resource, yet may have a lower reliability. Figure 1 classifies the different resources, which will be discussed below, by 3 of the 6 itemized criteria (development, data, area). Resources with gray background shades indicate domain-specific resources. The star in the top right corner positions our own approach.

Linguistic resources that focus on concept data and lexicographic relations are commonly called *thesauri*, *semantic word nets* or *lexicographic databases*. They typically comprise synonym, hypernym, meronym and cohyponym relations. Resources that provide information about entities (persons, locations, companies, countries etc.) are commonly called *knowledge bases* and can comprise much more specific relations (like *was born in*, *is located in*, *was founded in/by* etc.). In the remainder of this section, we first discuss manually created resources, then analyze different

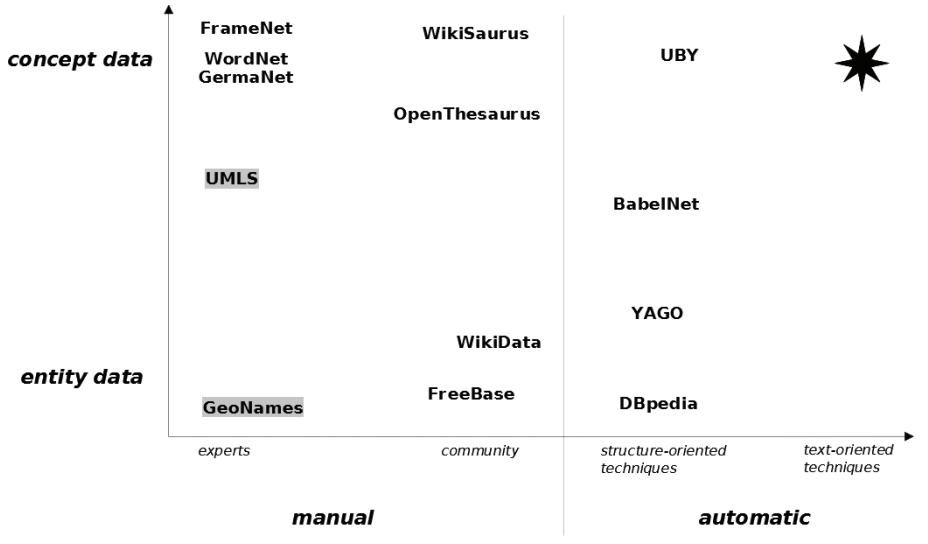

Fig. 1. Classification of selected background knowledge resources.

possibilities to exploit the web as background knowledge source and finally come to approaches that use Wikipedia as their primary source.

2.1. *Manually created resources*

One of the oldest and most popular linguistic resources is **WordNet**,[a] which has its roots in the mid-1980s.[24] Its content is manually derived by linguists, making it a highly precise resource. However, progress is relatively slow and WordNet lacks many modern terms, e.g., *netbook* or *cloud computing*. WordNet arranges words in so-called synsets, which are well-defined mental concepts having a specific sense. Words can point to one or several synsets and synsets can be referenced by one or several words. Currently, WordNet defines 82 115 noun synsets (concepts) and 117 798 nouns. This makes it an extensive source, although the general English language is believed to comprise up to a million words even without specific scientific terms.

GermaNet[b] is the German counterpart of WordNet, which provides a linguistic classification for most German nouns, verbs and adjectives. **EuroWordNet**[c] is a framework and thesaurus for multiple languages. Based upon the WordNet data structure, it was enhanced by a top-ontology serving as a semantic framework for the different languages. Currently, eight European languages have been integrated in this framework.

[a]http://wordnet.princeton.edu/
[b]http://www.sfs.uni-tuebingen.de/GermaNet/
[c]EuroWordNet

FrameNet is a different approach of organizing lexicographic items.[d] Instead of synsets, it defines so-called semantic frames describing a specific process, situation or event. For instance, the semantic frame "transfer" describes that there must be a person A (donor) giving some object B to a person C (recipient), and that this frame is activated by verbs like *to transfer*, *to give* etc. Semantic frames are related with each other, e.g., the semantic frame "Committing_crime" leads to the frame "Crime_investigation".[10]

Crowd sourcing is a promising approach to speed-up the laborious development of a comprehensive thesaurus by utilizing a community of volunteers. An exemplary effort is **OpenThesaurus** (German language thesaurus). As the contributors are no linguistic experts, we discovered that the precision is slightly below WordNet, though, and that a considerable amount of entity data is also incorporated (German cities, politicians, etc.). A smaller effort is **WikiSaurus**, a sub-project of the English Wiktionary providing synonyms, hypernyms, hyponyms and antonyms for selected concepts (while meronyms and holonyms are rare).[e] It currently provides some thousands of categories, though recent activity seems rather low and no API is applicable so far. **WikiData** is a collaboratively generated knowledge base about facts and entity data (like birth dates of persons). It also provides some concept data for categorization (e.g., *breast cancer* is a subclass of *cancer*, which again is a subclass of *disease*), thus partly combining the features of knowledge bases and thesauri.[f] **Freebase** is a large collaboratively generated knowledge base similar to WikiData, yet focuses more on the semantic web and machine readability.[7]

UMLS[g] is a large domain-specific knowledge base and thesaurus for the biomedical domain. It combines the vocabulary of various medical dictionaries and taxonomies in the so-called MetaThesaurus. A Semantic WordNet is used to classify terms and link them by a large amount of (biomedical) relations.[6] **GeoNames** is another domain-specific knowledge base, focusing on geographic data like locations, countries, rivers etc. It was developed out of a various amount of geographic ontologies and classifications.[h]

2.2. *Knowledge extraction from the web*

The development of large repositories with some millions of elements and relationships is only feasible with automatic approaches for knowledge acquisition from existing text corpora and especially from the web. This can either be done by directly extracting knowledge from documents and web content (e.g., Wikipedia) or by exploiting existing services such as web search engines. The latter approach is followed in Ref. 17, where a search engine is used to check the semantic relationship between

[d]https://framenet.icsi.berkeley.edu/fndrupal/
[e]http://en.wiktionary.org/wiki/Wiktionary:Wikisaurus
[f]http://www.wikidata.org
[g]http://www.nlm.nih.gov/research/umls/knowledge_sources/metathesaurus/index.html
[h]http://www.geonames.org/

two terms A and B. They send different phrases like "A is a B" (like "a computer is a device") or "A, such as B" (like "rodents, such as mice") to a search engine and decide about the semantic relation based on the number of returned search results and by analyzing the returned result snippets. Such an approach is typically not scalable enough to build up a repository, since the search queries are rather time-consuming and since there are typically restrictions in the allowed number of search queries. However, such approaches are valuable for verifying found semantic correspondences, e.g., for inclusion in a repository or for ontology mapping.

In Ref. 34 the authors use an ontology search engine called Swoogle to find background knowledge ontologies from the web for a specific mapping scenario. Such an approach faces the difficulty to find relevant ontologies. Furthermore, different resources may return inconsistent or even contradicting results, e.g., one resource suggesting a subset relation while the other resource suggests disjointness.

2.3. *Knowledge extraction from Wikipedia*

Numerous research efforts aim at extracting knowledge from Wikipedia, as a comprehensive and high quality (but textual) web information source and lexicon. The focus and goals of such efforts vary to a large degree. Examples include approaches that extract generalized collocations,[11] computing semantic relatedness between concepts or expressions[12,36] and word sense disambiguation.[26] More related to our work are previous efforts to derive structured knowledge and ontologies from Wikipedia, for example DBpedia, Yago and BabelNet.

We differentiate two main types of approaches for extracting knowledge from Wikipedia (or similar sources) which we call *structure-oriented* and *text-oriented* extraction. The first type exploits the document structure of Wikipedia articles such as info boxes, article headings and sub-headings and the Wikipedia-internal category system typically allowing a rather precise information extraction. This approach is followed by DBpedia, Yago and related projects. By contrast, text-oriented approaches works on the actual text content of Wikipedia articles and are thus based on natural language processing (NLP) and text mining methods. These approaches tend to be more complex and error-prone than structure-oriented ones. However, they are also able to obtain more detailed and more comprehensive information.

DBpedia[4] focuses on the extraction of structured content from info boxes in Wikipedia articles which is generally easier than extracting content from unstructured text. The extracted knowledge is mostly limited to named entities with proper names, such as cities, persons, species, movies, organizations etc. The relations between such entities are more specific (e.g., "was born in", "lives in", "was director of" etc.) than the linguistic relation types between concepts that are more relevant for ontology mappings and the focus of our work.

The *Yago* ontology[37] enriches DBpedia by classifying Wikipedia articles in a thesaurus, as the Wikipedia-internal categories are often quite fuzzy and irregular.

Yago thus contains both relations between entities, e.g., "Einstein was a physicist", as well as linguistic/semantic relations, e.g., "physicist is a scientist". The latter relations are derived by linking Wikipedia articles from category pages to the WordNet thesaurus. We experimented with Yago, but found that it is of relatively little help if WordNet is already used, e.g., Yago will not link concepts A and B if neither is contained in WordNet.

BabelNet contains millions of concepts and linguistic relations in multiple languages.[25] It utilizes mappings between Wikipedia pages and WordNet concepts as well as background knowledge from the SemCor corpus. Its precision is around 70–80%, depending on the language. The more recent *Uby* is a multilingual infrastructure for lexicographic resources integrating concepts from different sources such as WordNet, GermaNet, FrameNet, Wiktionary and Wikipedia. It comprises more than 4.2 million lexical entries and 0.75 million links that were both manually and automatically generated (using mapping algorithms).[16] Both BabelNet and Uby are useful resources, although they still restrict themselves to concepts and entities already listed in the existing sources. We aim at a more general approach for extracting semantic concept relations from unstructured text, even for concepts that are not yet listed in an existing repository such as WordNet.

2.4. *Text-oriented approaches*

Text-oriented approaches are used to extract information from textual resources, which is generally more challenging than information extraction from structural data. In 1992, Marti A. Hearst proposed the use of lexico-syntactic patterns to extract synonym and hyponym relations in unrestricted text, like "A is a form of B" (A is-a B) or "$A_1, \ldots, A_{n-1}$ and other A_n" ($A_1, \ldots, A_n$ are synonyms).[20] In Ref. 21, such *Hearst patterns* are used to create ontologies from Wikipedia pages. The approach focuses on the biological domain and can handle only simple semantic patterns. They obtain a rather poor recall (20%) but excellent precision (88.5%).

In Refs. 33 and 32, Ruiz-Casado and colleagues apply machine learning to learn specific Hearst patterns in order to extract semantic relations from Simple Wikipedia[i] and link them to WordNet. They only consider links between nouns that are Wikipedia entries (thus occurring as hyperlinks in the text), but in many cases relations are also between non-hyperlinked words. As they only link words (nouns) to WordNet concepts, they are facing the same coverage problem as mentioned for Yago. Simple Wikipedia has a quite restricted content, leading to only 1965 relationships, 681 of which are already part of WordNet. Snow *et al.*[35] also apply machine learning to learn Hearst patterns from news texts in order to decide whether words are related by hypernyms or hyponyms. In Ref. 13, the authors introduce a supervised learning approach to build semantic constraints for part-of relations in natural text. Those patterns are retrieved by using a selection of

[i]http://simple.wikipedia.org

WordNet part-of relations as training data, which are gradually generalized and disambiguated.

Sumida and Torisawa focus on finding hyponymy relations between concepts from the Japanese Wikipedia.[38] They exploit the internal structure of Wikipedia pages (headings, sub-headings, sub-sub-headings etc.) together with pattern matching and different linguistic features. They could retrieve 1.4 million relations with a precision of about 75%. Ponzetto and Strube[27] also exploit the category system and links of Wikipedia to derive is-a and non is-a relations by applying lexico-syntactic pattern matching.

In our approach, we will also apply semantic patterns to determine semantic relations similar to the previous approaches. However, we focus more on the actual text of Wikipedia articles (especially Wikipedia definitions) rather than on the existing category system, info boxes or hyperlinks between pages. Also, we are especially interested in conceptual relations (as opposed to links between named entities) and try to cover not only hyponym (is-a) relations, but also equal, part-of and has-a relations.

3. Semantic Relation Patterns

Semantic relation patterns are the core features in our approach to find semantic relations. We focus on their identification in the first sentence of a Wikipedia article which mostly defines a concept or term and thus contains semantic relations. The sample sentence in Fig. 2 contains two semantic patterns defining "ice skates". In this section, we introduce the notion of semantic patterns and discuss different variations needed in our approach. In the next section, we describe in detail the use of semantic patterns for finding semantic relations.

A semantic relation pattern is a specific word pattern that expresses a linguistic relation of a certain type (like hyponym resp. is-a). It connects two sets of words X and Y appearing left and right of the pattern, much like operands of a comparison relationship. There are general patterns for hyponym (is-a) relations, meronym (part-of) relations, holonym (has-a) relations and synonym (equal) relations, the is-a patterns being the most commonly occurring ones in Wikipedia definitions. For example, the simple pattern "is a" in "*A car is a wheeled motor vehicle.*" links the concepts *car* and *vehicle* by a hyponym relation. Having these two concepts and the

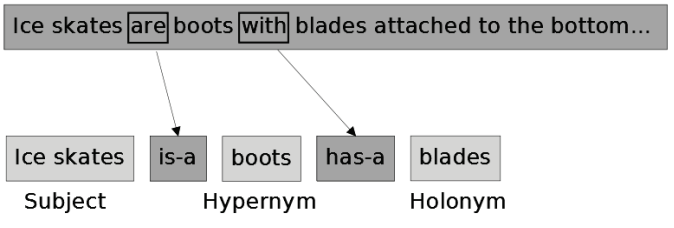

Fig. 2. Sample sentence containing two semantic relation patterns.

Table 2. Typical patterns for is-a relations (hyponyms).

Hypernym Patterns
is a
is typically a
is any form of
is a class of
is commonly any variety of
describes a
is defined as a
is used for any type of

semantic relation pattern, we can build the semantic relation (car, is-a, vehicle). The example in Fig. 2 shows that there may be more than one semantic pattern in a sentence that need to be correctly discovered by our approach.

3.1. *Is-a patterns*

According to our experiences, "is-a" patterns occur in versatile variations and can become as complex as "X is any of a variety of Y". They appear often with an additional (time) adverb like *commonly*, *generally* or *typically* and expressions like *class of*, *form of* or *piece of*, which are called collectives and partitives. They can appear in plural and singular ("is a" or "are a") and come with different determiners (like is a/an/the) or no determiner at all as in the *ice skates* example. They invariably come with a verb, but are not necessarily restricted to the verb *be*. Table 2 shows some examples of frequently occurring is-a patterns that we use in our approach. The list of patterns is extensible so that a high flexibility is supported.

3.2. *Part-of/has-a patterns*

Typical patterns for part-of and has-a relations are shown in Table 3. The adverb *within* and the prepositions "in" and "of" often indicate part-of relations, e.g., for "*A CPU is the hardware within a computer*", leading to (CPU, part-of, computer), and for "*Desktop refers to the surface of a desk*", leading to the correct relation

Table 3. Typical patterns for part-of relations (meronyms) and has-a relations (holonyms).

Meronym Patterns	Holonym Patterns
within	consists/consisting of
as part of	having
in	with
of	

Table 4. Typical synonym patterns in itemizations.

Synonyms Patterns
A, B and C
A, also called B
A, also known as B or C
A, sometimes also referred to as B

(desktop, part-of, desk). However, these patterns can also be misleading, as such prepositions can be used in various situations, as "*Leipzig University was founded in the late Middle Ages*", which would lead to the not really useful relation (Leipzig University, part-of, Middle Ages). Similar arguments hold for holonym patterns, where *consisting of* is often more reliable than the rather diversely used words *having* and *with*. Valid examples include "*A computer consists of at least one processing element*", leading to (processing element, part-of, computer) and the ice skates example resulting in (blades, part-of, ice skates). On the other hand, "*A screw-propelled vehicle is a land or amphibious vehicle designed to cope with difficult snow and ice or mud and swamp.*" is a misleading case, as it can lead to relations like "*snow*, *part-of*, *screw-propelled vehicle*".

3.3. *Equal patterns*

Finally, Table 4 shows some constructions for synonym relations. In itemizations occurring before another semantic pattern, the terms they comprise are generally synonyms (as in "*A bus* (*archaically also omnibus*, *multibus*, *or autobus*) *is a road vehicle*"). Outside itemizations, there are also a few binary synonym patterns like "is a synonym for", "stands for" (in acronyms and abbreviations) or "is short for" (in shortenings). They are quite rare in Wikipedia, as synonym words are typically comprised in exactly one page (for example, there is only one Wikipedia page for the synonym terms *car*, *motor car*, *autocar* and *automobile*). Thus, instead of a definition like "*A car is a synonym for automobile*" articles rather look like "*An automobile*, *autocar*, *motor car or car is a wheeled motor vehicle* [...]". In this case, four synonym terms are related to one hypernym term (*wheeled motor vehicle*). Our approach is able to identify multiple semantic relations in such cases.

4. Discovering Semantic Concept Relations

This section outlines in detail how we extract semantic concept relations from Wikipedia. The overall workflow is shown in Fig. 3. We start with a preparatory step to extract all articles from Wikipedia. For each article we perform the following six sub-steps:

(1) We check whether it is a relevant article for our repository (if not, we skip the article).

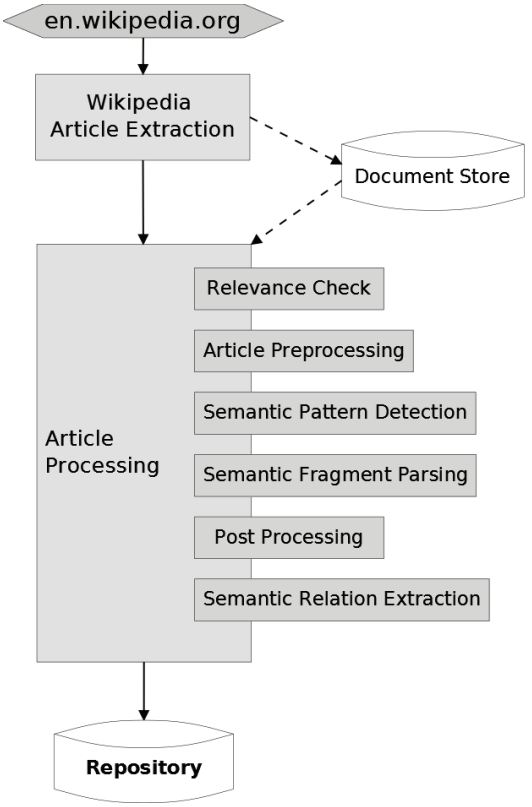

Fig. 3. Workflow to extract semantic relations from Wikipedia.

(2) We perform some preprocessing to extract its first sentence (the "definition sentence") and to tag and simplify this sentence.
(3) In the definition sentence, we identify all semantic relation patterns. If there are n such patterns ($n \geq 1$), we split the sentence at those patterns and thus obtain $(n + 1)$ sentence fragments. If there is no pattern, we skip the article.
(4) In each sentence fragment, we search for the relevant concepts that are linked by the semantic relation patterns.
(5) We perform some post-processing on the extracted information, e.g., word stemming.
(6) Having the terms and patterns, we build the respective semantic relations and add them to our repository.

The workflow is carried out automatically, i.e., no human interaction is required. It uses a few manually created resources, like a list of typical English partitives (e.g., *kind of*, *type of*, *genus of*), collectives (e.g., *class of*, *number of*, *range of*) and anchor terms for the pattern detection, but apart from that it does not need any additional background sources.

For our example sentence of Fig. 2, we would after preprocessing identify in the second step the two semantic patterns "is-a" and "has-a" and determine three sentence fragments. We also find out that *ice skates* is the subject (left operand) for both semantic relations while the other fragments refer to the second operand (object) for the relations. The fragments are further processed in the third step where we determine that the hypernym concept is *boots* and the holonym concept is *blades*. We finally derive the two semantic relations (ice skates, is-a, boots) and (ice skates, has-a, blades) and store them in the repository.

In the following, we first describe the preparatory extraction of articles from Wikipedia and then outline the six major steps to be performed per article.

4.1. *Extracting Wikipedia articles*

The easiest way to access the full Wikipedia content is to download the Wikipedia dump, which is basically a single XML file containing all articles with the respective content and meta information (like page id, creation date, categories in which it occurs etc.).[j] This file contains about 11 million entries and has an overall size of 44 GB (unzipped). However, the English Wikipedia comprises only about 4.3 million articles (as of October 2013), as there are many additional pages listed in the Wikipedia dump which are no articles, like category pages, redirects, talk pages and file pages (images, charts etc.).

The first step of our approach is to extract each Wikipedia article name together with the abstract section of the article. We will carefully remove the aforementioned pages that do not represent Wikipedia articles. As it is partly difficult to determine the abstract of an article (which is the section occurring before the table of contents), and as some articles simply do not contain abstract and main section, we extract the first 750 characters in each article. The ratio behind this limit of 750 characters is that we are currently only parsing the first sentence of each Wikipedia article, which is typically a definition sentence containing the relevant information. We may try parsing the second or third sentence later on, because they occasionally contain some additional information, but currently do not intend to parse the full text of Wikipedia articles, so that the first 750 characters of each article will suffice for our purposes.

The extracted text is subsequently cleaned from Wikipedia-specific formatting commands using the Java Wikipedia API (Bliki engine),[k] and then page name and extracted text are stored as documents in a document database (MongoDB) for further processing.

Redirects

Our approach is also able to handle redirect pages contained by Wikipedia. A redirect page is a reference from a specific keyword K to the actual article to which

[j]http://en.wikipedia.org/wiki/Wikipedia:Database_download
[k]http://code.google.com/p/gwtwiki/

it refers (named K'). The relation between K and K' is generally *equal*. We found four types of redirects in Wikipedia:

- **Homonyms**, e.g., the words *tramcar*, *streetcar*, *trolley car* etc. are synonymous and refer to the Wikipedia article *tram*.
- **Shortenings**, e.g., *Merkel* redirects to *Angela Merkel* (though there are also further pages about that name).
- **Different spellings**, e.g., the word *color* is spelled *colour* in some parts of the world, thus *colour* redirects to the *color* article.
- **Common Misspellings**, e.g., the word *libary* refers to the *library* article and *Merckel* refers to *Merkel*.

We are actually not interested in the last type of redirects, but cannot distinguish the different types when we parse the XML file. Thus, we extract all redirects as equal-relations and use them in our repository. To our experiences, misspelling redirects have no negative impact on our repository after all.

We only use redirects that appear to be relevant for our repository. We do not keep redirects of concepts consisting of more than 5 words, as they are often book titles, movies, organizations etc. We also discard redirects that consists of letters and numbers or letters and special characters. With this, we collected 2.04 million redirect relations.

Using re-directs is the only way of retrieving semantic relations without any form of natural language processing, and is thus very easy. However, it is restricted to equal-relations, and we use it only as an additional feature to obtain our semantic relations from Wikipedia. The core of our work is described in the following subsections, where we parse sentences using FSMs and can discover all kind of semantic relations.

4.2. *Relevance check*

In our approach we intend to build a lexicographic repository (thesaurus) and not a knowledge database. For this reason, we are only interested in concepts, not in entities, which means that articles about persons, locations, movies, novels, music albums, organizations etc. are irrelevant for our purposes.

However, most articles of Wikipedia refer to such entity data. In Fig. 4 we show the distribution of articles in Wikipedia by selected topics; we developed this chart manually by analyzing 300 random articles. As the picture clearly shows, concept articles only make up about 7% of all Wikipedia articles (about 320 000 articles), though in many cases there is no clear demarcation between concepts and entities, e.g., in the case of biological or medical terms (like species, plants, diseases etc.). WordNet also contains a large amount of biological or medical terms, and even a few selected persons like *Albert Einstein* or *Martin Luther*. Thus, also articles from different categories might be important for our repository and only classic entities like persons, places, movies, songs, albums etc. are irrelevant for our ambitions after

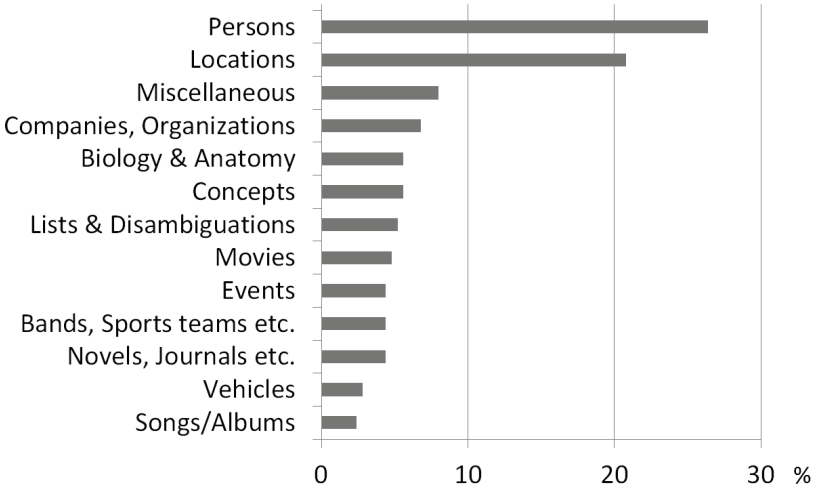

Fig. 4. Distribution of Wikipedia articles by selected subject.

all. This makes it hard to come up with an appropriate filter that blocks irrelevant information without blocking too much of the actual information.

A promising way to determine whether an article deals with concept or instance data is to analyze the categories in which the article appears. We made some interesting observations in this respect:

- Many articles about persons are in "Persons of `Country`" categories, and also in "`Year` births" and "`Year` deaths" categories.
- Many villages, towns and cities are in categories like "Towns in `Region/Country`", "Municipalities in `Region/Country`" etc.
- Many companies are in categories like "`Domain` companies" or "Companies of `Country`".
- Movies are often in categories like "`Year` movies".
- Novels are often in categories like "`Year` novels".
- Music albums are often in categories like "`Year` albums".

More specific property names, like sports clubs or national parties are more difficult to find. For instance, sports clubs are often listed in categories such as "German football clubs", implying the need of large lists of keywords to check a given category (as there are numerous sport fields, nations etc.). As a result we can only try to eliminate most entity articles, but will likely fail to identify all of them.

To effectively remove entity articles, we examined about 1000 random articles and analyzed their specific categories. We recognized that many categories clearly refer to entity articles, for instance *Rivers in Germany*, *Airlines established in 1980*, *2013 debut albums* etc. We manually developed regular expressions that are able to cover many of such categories and discard any article appearing in one of the

categories matching the regular expressions. We will call such categories *blocked categories*.

We denote the set of all Wikipedia categories by C_W and the set of categories in which a Wikipedia article occurs by C_A ($C_A \subseteq C_W$). Altogether, we created more than 400 regular expressions of the following types:

(1) **Exact match expressions (E_1):** An article is blocked if there is a $c \in C_A$ that matches exactly an expression $e \in E_1$, e.g., "*living people*" or "*given names*".
(2) **Starts-with expressions (E_2):** An article is blocked if there is a $c \in C_A$ that starts with an expression $e \in E_2$, e.g., "*rivers in **", "*airports in **", "*studio albums by **" etc.
(3) **Ends-with expressions (E_3):** An article is blocked if there is a $c \in C_A$ that ends with an expression $e \in E_3$, e.g., "** companies*" or "** organizations*".
(4) **Matches expressions (E_4):** An article is blocked if there is a $c \in C_A$ that matches an expression $e \in E_4$. In this case *match* refers to the matching of a regular expressions, e.g., "[1–2][0–9][0–9][0–9] births".

We developed 56 *exact match* expressions, 256 *starts-with* expressions, 43 *ends-with* expressions and 93 *match* expressions. Using the patterns, we tried to block anything except for concept articles and articles from the biomedical domain (species, anatomy, plants etc.), because our repository is intended to address this domain as well.

The general advantages that we get from the filtering techniques is:

- We keep the repository more compact, which will result in faster execution times for queries and a better scalability.
- We can prevent a couple of homonyms which are not necessary. For example, the word "autumn" not only refers to the season (concept), but also to movies, bands and albums. Not including such links may be in favor of the query precision.
- Building the repository becomes faster, since a great amount of articles does not has to be processed.

Though our filtering techniques appears rather strict, as an article is blocked as soon as a $c \in C_A$ matches any filter expressions, we discovered no notable removal of the essential concept articles. In Section 5.1 we will provide detailed information about recall and precision of the article filtering.

4.3. *Article preprocessing*

Before we start parsing the definition sentence of a Wikipedia article, we perform some textual preprocessing. We first replace intricate expressions that cannot be handled by our approach (*sentence simplification*). Such expressions will be replaced by simpler expressions or, in specific cases, removed entirely. For instance, we replace the rare and bulky expression "is any of a variety of" by "is any" which can be

Table 5. POS-tagging example for the *Ice skates* article.

Word	POS Tag	Word Class
Ice	NN	noun (singular)
skates	NNS	noun (plural)
are	VBP	verb (plural)
boots	NNS	noun (plural)
with	IN	preposition
blades	NNS	noun (plural)
attached	VBN	verb (past participle)
to	TO	"to"
it	PRP	personal pronoun

handled well by our approach. The advantage of such simplifications is that it avoids a more complex processing in later steps without losing information.

Secondly, we perform Part-of-Speech tagging (POS tagging) using the Apache OpenNLP Library for Java[1]. The POS tagger determines the word class of each word in the sentence and annotates the words accordingly. After this, the sentence "Ice skates are boots with blades attached to it." looks as follows:

Ice_NN skates_NNS are_VBP boots_NNS with_IN blades_NNS attached_VBN to_TO it_PRP.

Table 5 gives a clearer representation of the sentence together with the POS tags and their meaning.

4.4. *Identifying semantic relation patterns*

To identify semantic relation patterns, we parse the first sentence of an article word by word and apply a finite state machine (FSM) to discover these patterns. Fig. 5 shows a simplified version of the FSM for the is-a patterns, consisting of nine states. The dark gray states $(1, 2)$ represent initial states for different terms, so-called *anchor terms*, indicating the beginning of pattern and the entry point to the FSM. Altogether, we have two FSMs for the pattern detection, and thus two sets of anchor terms: One for is-a patterns, and one for part-of/has-a patterns. In the following, we will illustrate the gist of the is-a FSM.

Anchor terms can be in singular (like *is*, *describes*) or plural (like *are*, *describe*). If we find any anchor term in the sentence, we continue processing the FSM to which the anchor term refers (is-a or part-of/has-a). Starting from either of the two initial states, we check the following word or sequence of words in the sentence and can thus change into another state (transition). For instance, if the word after an anchor term is an adverb like "commonly", we change into state 3. If we reach

[1]http://opennlp.apache.org/

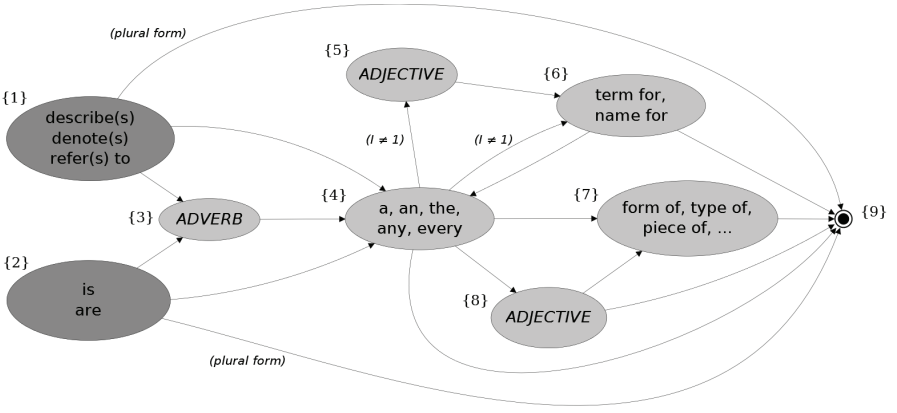

Fig. 5. FSM for parsing is-a patterns (simplified).

the final state (9), we have detected the entire pattern. We will then go on with the word-by-word parsing to look for another semantic relation pattern. For any newly found anchor term, we process the corresponding FSM until we have found all semantic patterns.

Some transitions have additional conditions, like $I \neq 1$, meaning that this transition can only be used if the initial state was not 1, or "plural form", meaning that this transition can only be used if the anchor term has the plural form. For example, "Ice skates **are** boots" uses this case, in which the state path is simply $(2, 9)$. If the verb is in singular, we normally have at least one determiner (covered by state 4). The states passed in the FSM can become more complex, as in the example "*A baker's rack* ***is a type of*** *furniture*", which has the state path $(2, 4, 7, 9)$ or "*Adenitis* ***is a general term for an*** *inflammation of a gland.*" $(2, 4, 5, 6, 4, 9)$.

Conditions for transitions into another state can become rather complex; they are mostly excluded from Fig. 5 for better legibility. In many cases we have to check the part of speech (like adjective, adverb, determiner) of the next word, or we have to check the next word as such. In state 4, for instance, there are two adjective states we could enter (5 or 8). Thus, to change into state 5, we have to check that the next word is an adjective, and the further two words are "term for" or "name for". We also have to check that the initial state has not been 1, as phrases like "A describes a term for B" are insensible, while "A is a term for B" is correct (initial state 2). If no transition is possible, we cannot reach the final state and leave the FSM. We then continue parsing the sentence to possibly find another anchor term.

The finite state machines we use were manually developed. We started with a basic implementation to handle simple patterns like A is a B, and then iteratively fine-tuned our approach to cover more specific cases. To determine these specific cases we processed several hundreds of randomly selected articles, searching for Wikipedia definitions that could not be processed. The revised FSMs are now able to handle most of the articles in Wikipedia.

Algorithm 1 Semantic pattern detection in a Wikipedia sentence

```
1: W ← List of words (input sentence)
2: A ← List of anchor terms
3: PatternList ← ∅
4: for all words w in W do
5:    Check whether w is an anchor term.
6:    if w is an anchor term a in A then
7:       Use FSM to which a refers
8:       Go through the FSM until a final state is reached
9:       if final state can be reached then
10:         Extract pattern covered by the FSM
11:         Add pattern to PatternList
12:         if size of PatternList = 2 then
13:            END
14:         end if
15:      end if
16:   end if
17: end for
```

Algorithm 1 provides additional details of the basic workflow of semantic pattern extraction. Its input are the list of words of the input sentence, a predefined list of anchor terms (as in the FSM) and an empty list of extracted patterns (which are sequences of words). We iterate through all words in the sentence until we reach its end (lines 4–17). If some word is an anchor term (line 5), we use the FSM to reach a final state as described above. If we find a transition to the final state (line 9), we have successfully discovered the semantic pattern, which is a sequence of words starting from the anchor term and ending at the word handled by the final state of the FSM. The pattern is added to the pattern list (line 11) and the iteration is continued if not at least two patterns have already been found (lines 12–14). The algorithm can thus determine up to two semantic patterns. The extraction of terms for the determined patterns is carried out in a further step described in section 4.5.

Most article definitions contain exactly one pattern, namely a hyponym (is-a) pattern. Sometimes, an additional meronym or holonym pattern is also available, generally succeeding the is-a pattern. Less frequently, no hyponym pattern is available, but a meronym or holonym pattern. We thus obtain mainly the following three combinations, where A, B and C are lists of terms:

(1) $A \xrightarrow{\text{Hyponym}} B$

(2) $A \xrightarrow{\text{Hyponym}} B \xhookrightarrow{\text{Hol./Mer.}} C$

(3) $A \xrightarrow{\text{Hol./Mer.}} B$

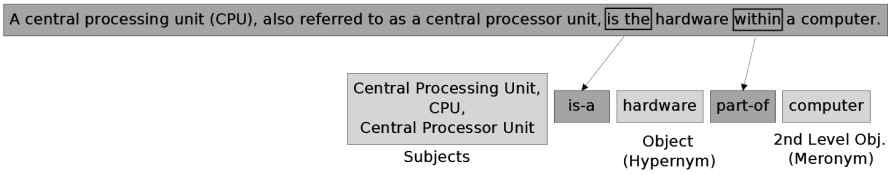

Fig. 6. Sample definition sentence with two patterns, 3 subject terms, 1 object term (hypernym) and 1 second-level object (meronym).

The hooked arrow in case 2 denotes that this pattern links A and C, but not B and C (as one might expect at first glance). Loosely related to the linguistic field of syntax, we call A the *subjects* and B the *objects* in the definition sentence (the patterns would be the verb then). If there is a C, we call C the *second-level objects* for disambiguation. Figure 6 gives an example of a Wikipedia sentence with two patterns, three subjects, one object and one second-level object. We split the sentence at the patterns and extract the subject, object and secondary object fragment for further processing in the next step.

If we find two relation patterns P and P' in a sentence, we use an individual threshold L (default value 7) specifying that at most L words may be between P and P'. If the number of words in between exceeds L, we reject P' and only use P. This strict regulation became necessary since we observed in test cases that patterns occurring in such a distance from the first pattern are frequently incorrect, e.g., meronym and holonym patterns for simple prepositions like "in" and "of".

4.5. *Parsing sentence fragments*

The sentence fragments representing subjects, objects or secondary objects need to be processed to identify the concept (term) or list of concepts that participate in a semantic relation. For this purpose, we apply a further FSM. In many cases, the nouns directly left and right from a semantic relation pattern represent already relevant concepts, thus allowing for a simple extraction. However, the following examples illustrate that such a strategy is generally too simple to correctly extract the relevant concepts:

(1) "*A wardrobe, also known as an armoire from the* ***French****, is a standing* ***closet****.*" (French is a closet)
(2) "*Column or pillar in* ***architecture*** *and* ***structural engineering*** *is a* ***structural element****.*" (architecture and structural engineering are structural elements)

The first example contains some additional etymological information, which can impair the extraction of subject terms. The second example contains a *field reference* that describes in which (scientific) field or domain the article is used, or if it is a homonym, to which field or domain the current page refers to. There is no general relation between field terms and semantic relation patterns, but between field terms

and subject terms. Thus, a subject term is normally "found" in the specified field or domain, which suggests the part-of relation. In example 2, we would thus conclude "column and pillar are part of architecture and structural engineering". Therefore, we extract both field references and subject terms.

It is especially important to retrieve all relevant concepts, which is difficult as some appear in additional parentheses (where also irrelevant terms may be found) or appositions. The FSM to process a single sentence fragment is therefore rather voluminous. The subject fragment parsing FSM alone contains about 20 states (5 initial states) and more than 40 transitions. There are subordinate FSMs that take control over specific tasks, such as to decide in an itemization of concepts where to start and where to stop, and whether an adjective or verb is part of a compound concept (like *high school*) or whether it is a word that does not belong to the actual concept (like a *wheeled vehicle*, in which *wheeled* is not part of the actual concept we would like to extract), although this is not always unequivocally possible to decide.

Parentheses are quite difficult to handle. It would be rather easy to simply remove all parentheses expressions, as they often contain only augmenting information, but can lead to insensible extractions or, more likely, bring the FSM into an illegal state. On the other hand, parentheses often contain synonyms which are very important for our purposes. We thus decided to run our approach in different configurations for every article. We first try to parse it without touching the parenthesis expression. If the article cannot be successfully parsed, we replace the parenthesis by commas and turn them into a real apposition. We also have a similar configuration in which the left parenthesis is replaced by ", or" and the right parenthesis by a comma. For instance, "*An auto* (*automobile*) *is a* ..." would be converted into "*An auto*, *or automobile*, *is a* ...", which is an expression the FSM can easily handle. Finally, if the parsing still fails, we remove the entire parenthesis expression. We risk to miss some synonyms then, but may be able to successfully parse the sentence after all.

Example

A simplified version of the subject fragment FSM is illustrated in Fig. 7 (we removed less important states and most of the transition conditions). As an example, we show how the subject fragment parsing works on the *Spindle* article: "*A spindle*, *in furniture and architecture*, *is an cylindrically*, *symmetric shaft*, [...]". Respectively, the subject fragment is "*A spindle*, *in furniture and architecture*", which will be passed to the subject fragment parser that uses the FSM. By stark contrast to the semantic pattern detection, the fragment-parsing FSMs start at the first word of the fragment. If the first word does not match any initial state, parsing the fragment is not possible.

In the FSM illustrated above, the states 1, 2, 3, 4 are initial states. Additionally, the FSM has two specialized states 4 and 7. State 4 handles concept terms in the

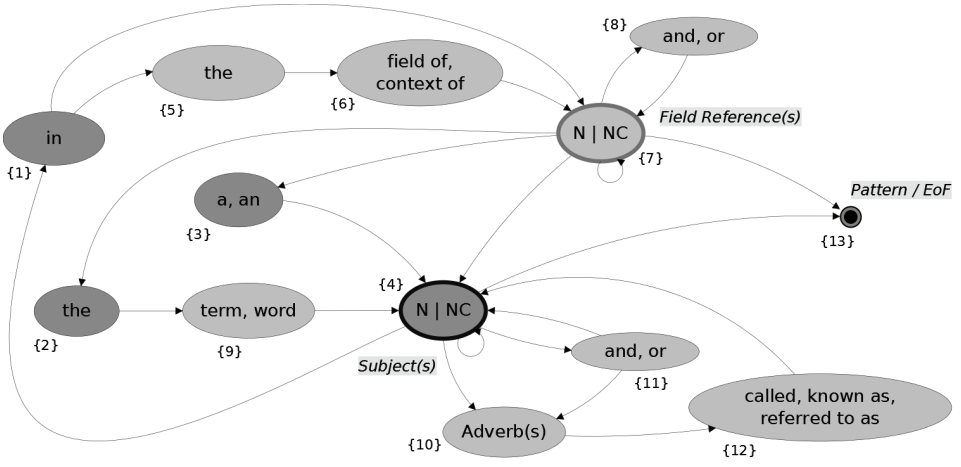

Fig. 7. FSM for parsing subject fragments (simplified).

fragment, while state 7 handles field references. Whenever such a state is reached, a subordinate algorithm is used to extract the concept at hand. In complex phrases, this algorithm has to decide where the concept name starts and where it ends (we discuss this difficulty in section 5.6).

In our example, the first word of the fragment is "*A*", so that we start at initial state 3. The next word *Spindle* is a noun, which takes us to state 4. As explained before, state 4 handles concepts, i.e., spindle would be considered a subject concept. The next word *in* gets us to state 1, following the word *furniture*, which gets us to state 7 (as *furniture* is a noun). Since state 7 handles field references, *furniture* will be considered a field reference. The next word *or* gets us to state 8 and then, with the next word being *architecture*, back to state 7; *architecture* is another field reference. Now having reached the end of the fragment, we follow the transition to final state 13 and have accomplished parsing the subject fragment. We successfully extracted *spindle* as a noun and *architecture* and *furniture* as the fields to which the word *spindle* refers.

4.6. *Post-processing*

Post-processing is necessary to obtain a high-quality repository without redundant or moot concepts. In this phase we remove remaining special characters on words, like commas, braces and additional or duplicate space characters. Then, we use a Java implementation of the Pling Stemmer to obtain the dictionary form (lemma) of each word. Even though the Pling Stemmer is quite a reliable word stemmer, and even though the English language is quite regular in its grammar, we observed that very occasionally a false stem is computed. For instance, the word *houses* is trimmed to *hous*, which is known as *overstemming* in computational linguistics. The same error is produced for words like *blouse* or *grouse*, which are among the

few words in English that end in *-ouse* and use the regular plural form. We may deal with this problem by adopting the algorithm, e.g., by handling such words manually, but may not be able to cover all special cases.

4.7. *Determining semantic relations*

Once the subject terms, object terms and field terms have been extracted, we build the semantic relationships. The outcome of each successfully parsed article is a set of (1:1)-relationships in which two terms are related by one semantic relation. There is one important aspect: The subjects are all related to the objects by the semantic relation patterns, but as they are synonyms, they are also related to each other by an equal relation. Hence, the sentence "*An automobile*, *autocar*, *motor car or car is a wheeled motor vehicle* [...]" results into four is-a relations as well as six equal relations for the four synonyms.

The equal relation does generally not hold between different objects as the following example shows: "*A rocket is a missile, spacecraft, aircraft or other vehicle that obtains thrust from a rocket engine.*" Although the four objects *missile*, *spacecraft*, *aircraft* and *vehicle* are all related to each other, they are not truly equivalent. This is a typical situation so that we do not derive equal relations between different object terms.

Let $|S|$ be the number of subjects, $|O_1|$ the number of objects and $|O_2|$ the number of second-level objects. The number of synonym relations R_s is:

$$|R_s| = \binom{|S|}{2} = \frac{|S| * (|S| - 1)}{2}\,. \tag{1}$$

The number of relations between S and O is $S * O$, since any subject is related to any object. The same holds for the number of relations between S and F (the field references). We thus have the following number of one-to-one relations $|R|$:

$$|R| = \binom{|S|}{2} + (|S| * |O_1|) + (|S| * |O_2|) + (|S| * |F|)\,. \tag{2}$$

Note that this number can become rather large. For instance, for 5 subject terms, 2 hypernym objects, 2 meronym objects and 1 field reference, we obtain 35 semantic relations from just one Wikipedia article. Although this is a rare case, it illustrates the richness of our strategy compared to previous approaches that only link Wikipedia page names with other resources like WordNet, leading to at most one link per Wikipedia article.

All determined semantic relations are finally added to a repository. They can then be used as background knowledge, e.g., for semantic ontology matching.

5. Evaluation

In our evaluation we analyze the effectiveness of the proposed approach for four different subsets of Wikipedia covering different subjects and domains. We first analyze the effectiveness of the relevance check (article filtering). We then evaluate

Table 6. Number of filtered articles.

	Before Filtering	After Filtering	Remains
Number of Concepts	4 386 119	1 051 170	24.0%
Number of Relations	12 519 916	2 843 428	22.8%

the different substeps to determine the semantic relation patterns and to determine the subjects and objects as well as the overall effectiveness regarding semantic relations. In the following, we first describe the used benchmark datasets. We then focus on the effectiveness of the FSMs to parse articles for finding semantic relation patterns. In subsection 5.4 we analyze the effectiveness of concept extraction to identify subjects, objects and secondary objects. We then evaluate how many concepts (subjects, objects and field references) could be retrieved from the Wikipedia articles and how many of them were correct. Subsequently, we evaluate how many relations we could generate from the patterns and concepts extracted, and how many of them were correct. We close this chapter with some observations on problem cases that could not yet be handled sufficiently.

5.1. *Article filtering*

We first analyze how many entity concepts can be blocked using the category filtering techniques described in section 4.2. Table 6 shows that our approach blocks more than three quarters of the concepts and relations it would extract without any relevance check. We retain about a million articles as likely concepts which is still substantially more than our sample-based estimate of about 320 000 concept articles in section 4.2 which does not, however, include the concepts from the biomedical domain. We analyzed 500 random articles from the remaining set of articles and checked for each whether it describes a concept or entity. We observed that the number of concepts is only about 51%, while about 49% are still entities. These remaining entities were often geographic places (like villages) that were not in any category regarded by our filter expressions. In fact, the majority of Wikipedia articles seems to be very specific often referring to only one category. Further entities that were not blocked are from specific subdomains of wars, contracts or company products, such as *Casio FA-6*, which is a calculator developed by Casio. Though there is a category "Casio calculators" which we could handle with an exact match expression or an ends-with expression "* *calculators*", it seems impossible to manually cover all such special cases, given the huge number of Wikipedia categories. Hence, the applied category blocking is effective but inherently unable to eliminate all entity pages.

In addition to the recall, we also analyzed the precision of our filtering technique. We checked 800 random articles that were blocked and found only one single article which we would consider a concept. It was the article "Bishop to the Forces", referring to a specific office of the Anglican church, which we rather considered a

concept and not an entity (while the incumbents of this office would be entities). This means that 99.9% of all filtered articles were true entity concepts.

Having roughly 50% of entities in our extraction result, we see a clear need for further work to improve article filtering. Still the proposed approach can also determine valid relations for entities and these relations will likely not impair much ontology or schema matching applications. This is because these applications only ask for relations between concepts so that the entity relations typically will not come into play. For instance, if we compare two vehicle taxonomies and want to know whether there is a relation between *convertible* and *car*, and of which type this relation is, the only thing that matters is that there is a relation (*convertible*, *is-a*, *car*) in the repository. Possible entity relations such as (*BMW i5*, *is-a*, *car*) or (*Porsche Carrera 4S Cabriolet*, *is-a*, *convertible*) are unnecessary but will not impact queries as the one mentioned above. We will further investigate this issue in future work and determine the impact of entity relations on mapping results.

5.2. *Benchmark datasets*

To evaluate our approach we have chosen four sets of Wikipedia articles as benchmark datasets, which we created manually as there are no general, type-specific benchmarks for our purpose available. Each such article set consists of all articles in a specific category, with the exception of "List of" articles that we neglected (as they never contain any semantic relations). We tried to use categories from different domains and with a representative number of articles W. We aimed at benchmarks containing around 100 articles, so that we were able to manually create the perfect result within a feasible amount of time, which includes to specify the exact number of subject and objects concepts, fields references and patterns for each benchmark article. Categories often contain sub-categories, which we did not include, though, thus obtaining moderate benchmarks containing between 94 and 186 articles.

Wikipedia categories are rather heterogeneous, which makes the benchmark datasets quite interesting. For instance, in the furniture category there are both very general concepts (like *couch*, *desk*, *seat*) and specific concepts (like *cassone*, *easel*, *folding seat*). By contrast, some general concepts one would definitely expect in the furniture category are not listed there, such as *chair*, *table* or *bed* (although *Windsor chair*, *sewing table* and *daybed* are listed). Typically, those concepts have a separate sub-category then.

Table 7 gives an overview of the datasets we use in our evaluation. The datasets *furniture*,[m] *infectious diseases*[n] and *optimization algorithms*[o] refer to category pages while *vehicles*[p] is based on an outline page (which is similar to a list, but represented as a Wikipedia article). The datasets were generated in October 2013 and

[m]http://en.wikipedia.org/wiki/Category:Furniture
[n]http://en.wikipedia.org/wiki/Category:Infectious_diseases
[o]http://en.wikipedia.org/wiki/Category:Optimization_algorithms_and_methods
[p]http://en.wikipedia.org/wiki/Outline_of_vehicles

Table 7. Benchmark datasets with their number of articles W and number of parsable articles W_p.

Name	Domain	W	W_p
Furniture (F)	General	186	169
Infectious Diseases (D)	Medicine	107	91
Optimization Algorithms (O)	Mathematics	122	113
Vehicles (V)	General	94	91

their articles may slightly differ from current versions due to some recent changes. Two benchmarks (*furniture*, *vehicles*) contain very general concepts while *infectious diseases* refers to the medical domain and *optimization algorithms* to the mathematical domain. The latter two benchmarks contain rather scientific articles and are more difficult to process.

It turned out that not all Wikipedia articles in the datasets could actually be used for our purposes, since they include articles that do not have a classic definition using a hypernym, holonym or meronym. These articles do not contain any specific semantic relation pattern and are thus not parsable. Examples of such non-parsable articles include:

- *Anaerobic infections are caused by anaerobic bacteria.*
- *Hutchinson's triad is named after Sir Jonathan Hutchinson (1828–1913).*
- *A diving chamber has two main functions: as a simpler form of submersible vessel to take divers underwater and to provide a temporary base and retrieval system* [...]

We exclude such articles from our evaluation and only consider the parsable articles W_p. The number of these articles is shown in the last column of Table 7.

5.3. *Article parsing and pattern discovery*

In this section, we evaluate how many articles we were able to fully parse using our FSMs. This includes the detection of at least one semantic relation pattern in the definition sentence, the sentence fragmentation and the extraction of at least one subject and one object. We use the classic recall and precision measures to evaluate our approach:

Given a set of parsable Wikipedia articles W_p, let ω be the number of articles we could successfully parse, and ω^T the number of articles where the correct semantic relation pattern was detected. We determine the recall (accuracy) for parsing r_{pars}, the recall for finding the correct semantic relation pattern r_{rel}, and the precision for finding semantic relation pattern p_{rel} as follows

$$r_{pars} = \frac{\omega}{W_p}, \qquad r_{rel} = \frac{\omega^T}{W_p}, \qquad p_{rel} = \frac{\omega^T}{\omega}. \tag{3}$$

Table 8. Evaluation of pattern detection.

	W_p	ω	ω^T	r_{pars}	r_{rel}	p_{rel}
F	169	148	142	0.88	0.84	0.96
D	91	80	80	0.88	0.88	1
O	113	84	84	0.74	0.74	1
V	91	87	87	0.96	0.96	1

Table 8 shows for each benchmark dataset the number of parsable articles (W_p), the number of parsed articles (ω), the number of correctly detected patterns (ω^T) as well as parsing recall r_{pars}, relation recall r_{rel} and relation precision p_{rel}. Our approach can parse between 74% and 96% (86% on average) of all Wikipedia articles that contain any semantic relations. Parsing the scientific articles in dataset O tends to be more error-prone (74%) than rather general articles (F, V) with 88–96% parsing recall.

The semantic patterns were in most cases correctly detected, leading to a precision of 96–100%. With the exception of only one article, the first pattern discovered in the definition sentence was invariably a hyponym pattern. If there was a second semantic pattern, it was invariably a holonym or meronym pattern.

5.4. *Term extraction*

Now that we have demonstrated how many articles could be parsed and in how many cases we detected the correct semantic pattern, we evaluate how many of the terms (subjects, objects and fields) encoded in the articles of the benchmark datasets were discovered (recall), and how many of the extracted terms were correct (precision). These terms, together with the semantic patterns (relations) make up the overall semantic relations we can derive from the Wikipedia article sets (we discuss these relations in the next subsection).

We denote T^P as the terms that occur in Wikipedia pages which were parsed by our approach. We denote T_C as the correctly identified terms and T_F as the falsely identified terms. Again, we use recall and precision to assess our approach:

$$r = \frac{T_C}{T^P}\,, \tag{4}$$

$$p = \frac{T_C}{T_C + T_F}\,. \tag{5}$$

Table 9 shows recall and precision of the term extraction. We provide results for all types of terms we extracted, i.e., subjects (S), objects (O), second-level objects (2L O) and field references (F). The recall is similarly good for all datasets and about 83–94% of the subjects and first-level objects could be correctly extracted. Extracting the second-level objects (has-a or part-of relations) is more difficult and ranges from 66–86%.

Table 9. Recall and precision for term extraction.

	r				p			
	S	O	2L O	F	S	O	2L O	F
F	0.93	0.90	0.66	0.8	0.95	0.87	0.58	0.73
D	0.85	0.87	0.70	1	0.96	0.80	0.57	0.67
O	0.84	0.91	0.81	1	0.88	0.88	0.36	0.92
V	0.83	0.94	0.86	—	0.96	0.94	0.49	—

Table 10. Number of extracted concepts and relations from each benchmark.

	W_P	Subj.	Obj.	2L O	Fields	Rel.
F	169	200	142	43	4	373
D	91	111	58	26	4	206
O	113	84	66	6	23	137
V	91	138	78	17	0	280

Precision is a little higher for subjects, where 88–96% of the extracted concepts where correct, while only 80–94% of the extracted first-level objects were correct. Extracting the second-level objects is more error-prone. We achieve only 36–58% precision, meaning that a considerable amount of terms are extracted which are actually no concept being part in any relation.

Field references occurred only scarcely in the considered benchmark datasets. There was no field reference in V, which is quite natural, as there is no "field of vehicles". In the other scenarios, we found most of the field references (80–100%) with a precision ranging between 67% and 92%.

Table 10 shows the number of articles in each benchmark and the number of terms we could correctly extract. The number of subjects is always highest, because many synonyms are found in the article definitions. The number of first-level objects is lower, as we find generally only one object (hypernym) in the definition. The number of second-level objects is again lower, as meronym and holonym relations occur only occasionally. The last column describes the number of correct relations we could derive from each benchmark. Comparing this number with the number of articles in the benchmark, it can be seen that we are able to extract an average amount of 1.2 to 3.1 relations per article (including articles we failed to process).

5.5. *Evaluation of semantic relations*

We have demonstrated how many semantic patterns we could correctly detect, and how many subject terms, object terms and field references we could extract. As a last step, we have to put these terms and patterns together to obtain a set of semantic relations. We will now show how many of these relations we could derive from each benchmark dataset, and how many of them were correct.

Table 11. Number of relations per benchmark, correctly extracted relations, falsely extracted relations as well as recall, precision and F-Measure.

	Rel. in W_p	Correct Rel.	False Rel.	r	p	f
F	497	373	87	0.75	0.81	0.78
D	323	206	67	0.64	0.76	0.69
O	182	137	49	0.76	0.74	0.75
V	413	280	66	0.68	0.81	0.74
$\sum$	1 415	996	269	0.70	0.79	0.75

Table 12. Distribution, recall and precision for each individual relation type in the furniture benchmark.

	Share	r	p	f
equal	24.1%	0.73	0.87	0.79
is-a	55.4%	0.87	0.88	0.87
has-a/part-of	19.4%	0.58	0.63	0.60
field ref.	1.1%	0.36	0.57	0.44
Average		75%	81%	78%

Table 11 presents these final results of our approach. It contains the number of overall relations contained per benchmark, as well as the number of correctly and falsely extracted relations with the corresponding recall and precision values. We can extract 64–76% of all relations that are in the benchmark with an average recall of 70%. Precision is slightly higher, ranging between 74% and 81% with an average of 79%. The F-Measure ranges between 69% and 78% with an average of 75%.

Eventually, we present the detailed results for each relation type in Table 12, which we performed on the Furniture benchmark. The first column specifies how often each individual relation type occurred (we computed the value on the correctly extracted relations). As it can be seen, more than half of all extracted relations are is-a relations. Equal-relations (24%) and has-a resp. part-of relations (19%) occur less often while field references are extremely rare (1%). Regarding the quality of each type, is-a relations have the best recall and precision, while equal-relations have a similar precision but somewhat lower recall; has-a resp. part-of relations only achieve moderate results both in recall and precision. Similar observations can be made w.r.t. the field references, although this value is difficult to judge, as there were only 4 field references in the benchmark.

At last, we show the number of falsely extracted concepts and falsely extracted relations in Table 13. Columns 2–5 show how many subject terms, object terms and field references were falsely extracted, while column 6 provide the overall number of falsely extracted concepts in each benchmark. The number of falsely extracted relations is shown in column 7.

Table 13. Number of falsely extracted concepts and relations per benchmark.

	S	O	2L O	F	$\sum$ Con	Rel
F	9	21	30	1	61	88
D	5	15	19	2	41	67
O	12	9	12	2	35	49
V	6	5	19	0	30	66
$\sum$	32	50	80	5	167	270

Falsely extracted concepts are the result of the term extraction step, while falsely extracted relations are the result of combining falsely extracted concepts with the semantic relation expressed by the detected pattern. As shown in the table, the number of wrong relations is larger than the number of wrong concepts since we can have multiple relations per concept. Altogether, 167 false concepts were extracted which resulted in 270 falsely extracted relations.

5.6. *Observations*

The evaluation results show that the proposed approach is already highly effective as it correctly parsed 74–98% of all parsable articles and retrieved approx. 70% of all relations in the 4 benchmarks with an approx. precision of 79%. Still, we were not able to detect all relevant relations, nor could we prevent erroneous extractions. To explain and illustrate the reasons for extraction errors we discuss some of the observed problem cases in the following.

Parsing errors are often caused by complex sentence structures, especially for complex introductions or subjects. Examples of articles that could not be successfully parsed include:

(1) *Cryptic Infections: an infection caused by an as yet unidentified pathogen ...*
(2) *A hospital-acquired infection, also known as a HAI or in medical literature as a nosocomial infection, is an infection ...*
(3) *Lower respiratory tract infection, while often used as a synonym for pneumonia, can also be applied to other types of infection including ...*

Example 1 does not contain any semantic relation pattern and uses the dictionary notation that is also used in Wiktionary but uncommon in Wikipedia articles. The two other examples have too complex subject fragments leading the parser into an illegal FSM state. A more advanced FSM might be able to handle these cases, but being very specific Wikipedia definitions, would have only little impact on the overall recall.

Regarding term extraction, recall problems are typically caused by complex expressions with parentheses that have to be removed in order to successfully parse the sentence. If such parentheses contain relevant terms, they will not be extracted.

Table 14. Distribution of relations extracted from Wikipedia.

Type	# Occurrences	Percentage
Equal	999 168	35.1%
Is-a	1 117 828	39.3%
Part-of/Has-a	726 432	25.6%

The POS-tagging is also error-prone, as in the following snippet "*A dog sled is a sled used for ...*", where both occurrences of *sled* are tagged as verbs, although the noun is meant. Our FSM does not expect a verb before or after the is-a pattern, so would refuse the article. In an extended version of our system, we also consider the page name of the article to be parsed. If a word in the sentence appears in the page name (as *sled* in the example is part of the article name *dog sled*), we accept the word even if it has an unreasonable word class. Still, this cannot avoid all abortions caused by erroneous POS-tagging.

Precision of term extraction is, among others, impaired by the following reasons:

- The parser draws much on the POS tags of the words in a sentence. In complex words, it may be misled, as in the following example: *A minibus is a passenger carrying motor vehicle.* The extracted relation is "A minibus is a passenger", as it does not discover the actual compound word "passenger carrying motor vehicle".
- Similarly, compounds cannot always be correctly determined. For instance, in "*A draisine is a light auxiliary rail vehicle*" the object "*light auxiliary rail vehicle*" is extracted. However, the actual compound would be *rail vehicle*, while *light auxiliary* is an attribute. The parser cannot generally ignore adjectives (and participles), as some compound words contain them (like *high school*, *bathing suite*, *pulled rickshaw*).
- Sometimes, itemizations contain misleading nouns as in "*Jet pack*, *rocket belt*, *rocket pack and similar names are used for various types of devices*", where "similar names are devices" is extracted. Similar arguments hold for expressions like "*is a noun for*", "*is the act of*" etc., which are all relatively rare. We will extend our list of simplifications to avoid such pitfalls.

We also compared our extracted results with the information encoded in the Wikipedia category system. Although many relations are found in the category system (like *table* is listed under the category of *furniture*), we were also able to extract is-a relations not provided by the category system. For instance, we extracted that *paint* is a *liquid*, but the paint article is only listed under the concept of paints, and we could not find any link between liquids and paints in the category system. Furthermore, the category system only provides is-a relations, while we can also extract has-a and part-of relations. Eventually, we can extract the various synonyms for a term described by an article, which can only partly be retrieved

using the redirect pages in Wikipedia, as such pages may not necessarily cover all synonyms found in the article definition.

6. Results on Overall Extraction and Usage of Semantic Relations

The execution of our approach on a Windows Server 2008 having 72 GB of RAM took about 8 hours. We needed about 40 minutes to parse and clean the Wikipedia XML dump file and about 2 hours 30 minutes to extract each article with the article abstract and store the information in MongoDB. The final processing of the roughly 4.3 million articles, with article filtering, took 4 hours and 56 minutes. This means that processing an article only requires about 4.1 ms and that our program can process about 244 Wikipedia articles per second or about 880 000 articles per hour. Since we process the Wikipedia articles consecutively and write the results in a text file, our approach has nearly linear time complexity (with the number of articles being the variable size).

According to the evaluation in the preceding section (Table 11), the precision of our relation extraction is roughly 80%. We thus have to expect that about 20% of the relations in the repository (about 569 000 relations) are false or irrelevant. Since false relations impair the quality of the repository, we plan a semi-automatic verification of all extracted relations in future work by utilizing the feedback of a web search engine. Similar to the approach presented in Ref. 17, we will transform a relation into a natural language expression and submit it to a search engine. If the search engine returns a representative amount of results, we accept this relation as valid; otherwise, we would reject it. For instance, for the semantic relation (mouse, part-of, computer), we would formulate search queries such as "*mouse is part of a computer*" or similar queries. This sample query, for instance, returns 14 results on Google and can thus be accepted as a reasonable relation. To cope with the common restrictions of search engines and the formulation of ideal queries is subject for future work.

We could already integrate the extracted Wikipedia relations together with relations from WordNet in a semantic repository called SemRep, which is now part of our ontology matching tool STROMA. STROMA[3] follows a so-called enrichment strategy where the correspondences found with a standard ontology matching tool (e.g., COMA or AgreementMaker) are enhanced by their correct semantic type. So far, STROMA mainly used WordNet as background knowledge, but can now also use the newly determined relations from SemRep. Given a correspondence between two matching concepts c_1, c_2, STROMA determines the relation type of the correspondence by its built-in strategies and the new SemRep repository.

To show how the new background knowledge can improve the mapping quality we performed an initial evaluation with three tests for matching product categories, two between Amazon and Ebay (furniture, groceries) and an additional one between Amazon and Zalando (clothing). The manually determined perfect mapping result consist of 136–169 correspondences. STROMA was used to determine the correct relation type for each correspondence without the Wikipedia relations and achieved

Table 15. Original F-Measure of STROMA and new F-Measure achieved with SemRep.

	Original F-Measure	New F-Measure	Change
Furniture	0.669	0.823	+0.154
Groceries	0.384	0.485	+0.101
Clothing	0.441	0.720	+0.279

F-Measures between 38.4% and 66.9%. As shown in Table 15, the usage of SemRep led to significantly better results, which are now between 48.5% and 82.3%. With the knowledge gathered from Wikipedia, the F-Measure could thus be increased between 10.1% and 27.9%, which demonstrates the usefulness and applicability of the relations extracted from Wikipedia according to the proposed approach.

7. Outlook and Future Work

We proposed and evaluated a novel approach to extract semantic concept relations from unstructured Wikipedia articles. The approach focuses on the analysis of the definition sentence of Wikipedia articles and uses finite state machines to extract semantic relation patterns and their operands to discover semantic relations. The approach is flexible and can find several semantic relations of different types (is-a, part-of, has-a, equal) per article. The evaluation showed the high effectiveness of the approach for different domains and on real mappings. By applying the approach to the entire Wikipedia, we could extract more than 2.8 million, still unverified, semantic relations for more than a million concepts.

We have integrated the extracted information in a new repository of semantic relations, combined with further resources and thesauri such as WordNet and UMLS. A preliminary evaluation already showed the high value of the dervied relations for improving ontology mappings.

In future work, we want to further improve the elimination of Wikipedia articles on entities to better focus on concepts and their relations. We also want to semi-automatically verify the determined relations with a search engine approach and thereby improve the quality of the new repository of semantic concept relations.

Acknowledgments

This study was partly funded by the European Commission through Project "LinkedDesign" (No. 284613 FoF-ICT-2011.7.4).

References

1. Zharko Aleksovski, Michel Klein, W. Ten Kate and Frank van Harmelen, Matching unstructured vocabularies using a background ontology, in *Proc. of the 15th Int. Conf. on Knowledge Engineering and Knowledge Management* (*EKAW'06*), eds. S. Staab and V. Svatek, No. 4248 in Lecture Notes in Artificial Intelligence (Springer, 2006), pp. 182–197.

2. Patrick Arnold and Erhard Rahm, Semantic enrichment of ontology mappings: A linguistic-based approach, in *Advances in Databases and Information Systems*, Vol. 8133 (Springer, 2013), pp. 42–55.
3. Patrick Arnold and Erhard Rahm, Enriching ontology mappings with semantic relations, *Data and Knowledge Engineering* (2014).
4. Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak and Zachary Ives, DBpedia: A nucleus for a web of open data, in *6th Int. Semantic Web Conference* (*ISWC*), LNCS 4825 (Springer, 2007), pp. 722–735.
5. Zohra Bellahsene, Angela Bonifati and Erhard Rahm (Eds.), *Schema Matching and Mapping* (Springer, 2011).
6. Olivier Bodenreider, The unified medical language system (UMLS): Integrating biomedical terminology, *Nucleic Acids Research* **32** (Database-Issue) (2004) 267–270.
7. Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge and Jamie Taylor, Freebase: A collaboratively created graph database for structuring human knowledge, in *Proc. of the 2008 ACM SIGMOD Int. Conf. on Management of Data* (*SIGMOD '08*) (ACM, New York, USA, 2008), pp. 1247–1250.
8. Jérôme David, Fabrice Guillet and Henri Briand, Association rule ontology matching approach, *International Journal on Semantic Web & Information Systems* **3**(2) (2007) 27–49.
9. Jérôme Euzenat and Pavel Shvaiko, *Ontology Matching*, 2nd edn. (Springer-Verlag, 2013).
10. Charles J. Fillmore, Christopher R. Johnson and Miriam R. L. Petruck, Background to FrameNet, *International Journal of Lexicography* **16**(3) (2003) 235–250.
11. Tiziano Flati and Roberto Navigli, Spred: Large-scale harvesting of semantic predicates, in *Proc. 51st Annual Meeting of the Association for Computational Linguistics* (2013), pp. 1222–1232.
12. Evgeniy Gabrilovich and Shaul Markovitch, Computing semantic relatedness using Wikipedia-based explicit semantic analysis, in *Proc. 20th Int. Joint Conf. on Artificial Intelligence* (2007), pp. 1606–1611.
13. Roxana Girju, Adriana Badulescu and Dan Moldovan, Learning semantic constraints for the automatic discovery of part-whole relations, in *Proc. Conf. North American Chapter of the Association for Computational Linguistics on Human Language Technology* (*NAACL '03*) (ACL, 2003), pp. 1–8.
14. Fausto Giunchiglia, Pavel Shvaiko and Mikalai Yatskevich, S-Match: An algorithm and an implementation of semantic matching, in eds. Christoph Bussler, John Davies, Dieter Fensel and Rudi Studer, *Proc. 1st European Semantic Web Symposium*, Vol. 3053 of LNCS (Springer, 2004), pp. 61–75.
15. Anika Groß, Júlio Cesar dos Reis, Michael Hartung, Cédric Pruski, and Erhard Rahm, Semi-automatic adaptation of mappings between life science ontologies, in *Proc. 9th Int. Conf. on Data Integration in the Life Sciences* (*DILS*), Lecture Notes in Bioinformatics (LNBI) 7970 (Springer, 2013), pp. 90–104.
16. Iryna Gurevych, Judith Eckle-Kohler, Silvana Hartmann, Michael Matuschek, Christian M. Meyer and Christian Wirth, Uby: A large-scale unified lexical-semantic resource based on LMF, in *Proc. 13th Conf. European Chapter of the Association for Computational Linguistics* (*EACL '12*) (ACL, 2012), pp. 580–590.
17. Willem Robert Van Hage, Sophia Katrenko and Guus Schreiber, A method to combine linguistic ontology-mapping techniques, in *Proc. Int. Semantic Web Conference* (*ISWC*), LNCS 3729, (2005), pp. 732–744.

18. Faycal Hamdi, Brigitte Safar, Nobal B. Niraula and Chantal Reynaud, Taxomap alignment and refinement modules: Results for OAEI 2010, in eds. Pavel Shvaiko, Jérôme Euzenat, Fausto Giunchiglia, Heiner Stuckenschmidt, Ming Mao and Isabel F. Cruz, *Proc. 5th Int. Workshop on Ontology Matching* (*OM*), CEUR Workshop Proceedings 689 (2010).
19. Michael Hartung, James F. Terwilliger and Erhard Rahm, Recent advances in schema and ontology evolution, in *Schema Matching and Mapping* (Springer, 2011), pp. 149–190.
20. Marti A. Hearst, Automatic acquisition of hyponyms from large text corpora, in *Proc. 14th Conf. on Computational Linguistics* (*COLING '92*) (ACL, 1992), pp. 539–545.
21. Aurelie Herbelot and Ann Copestake, Acquiring ontological relationships from Wikipedia using RMRS, in *Proc. ISWC 2006 Workshop on Web Content Mining with Human Language Technologies* (2006).
22. Yves R. Jean-Mary and Mansur R. Kabuka, ASMOV: Ontology alignment with semantic validation, in *Joint SWDB-ODBIS Workshop* (2007).
23. Sabine Maßmann, Salvatore Raunich, David Aumüller, Patrick Arnold and Erhard Rahm, Evolution of the COMA match system, in *Proc. 6th Int. Workshop on Ontology Matching* (*OM*), CEUR Workshop Proceedings 814 (2011).
24. George A. Miller, Wordnet: A lexical database for English, *Commun. ACM* **38**(11) (November 1995) 39–41.
25. Roberto Navigli and Simone Paolo Ponzetto, Babelnet: Building a very large multilingual semantic network, in *Proc. 48th Annual Meeting of the Association for Computational Linguistics* (*ACL '10*) (2010), pp. 216–225.
26. Simone Paolo Ponzetto and Roberto Navigli, Knowledge-rich word sense disambiguation rivaling supervised systems, in *Proc. 48th Annual Meeting of the Association for Computational Linguistics* (*ACL '10*) (2010), pp. 1522–1531.
27. Simone Paolo Ponzetto and Michael Strube, Deriving a large scale taxonomy from Wikipedia, in *Proc. 22nd National Conf. on Artificial Intelligence* (*AAAI'07*) (2007), pp. 1440–1445.
28. E. Rahm, Towards large scale schema and ontology matching, in *Schema Matching and Mapping*, Chap. 1 (Springer, 2011), pp. 3–27.
29. E. Rahm and P. A. Bernstein, A survey of approaches to automatic schema matching, *VLDB Journal* **10** (2001) 334–350.
30. Salvatore Raunich and Erhard Rahm, ATOM: Automatic target-driven ontology merging, in *Proc. 2011 IEEE 27th Int. Conf. Data Engineering* (*ICDE '11*) (2011), pp. 1276–1279.
31. Salvatore Raunich and Erhard Rahm, Target-driven merging of taxonomies with Atom, *Information Systems* **42** (2014) 1–14.
32. Maria Ruiz-Casado, Enrique Alfonseca and Pablo Castells, Automatic extraction of semantic relationships for wordnet by means of pattern learning from wikipedia, in *Proc. 10th Int. Conf. Natural Language Processing and Information Systems* (*NLDB'05*) (Springer, 2005), pp. 67–79.
33. Maria Ruiz-Casado, Enrique Alfonseca and Pablo Castells, Automatising the learning of lexical patterns: An application to the enrichment of WordNet by extracting semantic relationships from Wikipedia, *Data & Knowledge Engineering* **61**(3) (2007) 484–499.
34. Marta Sabou, Mathieu d'Aquin and Enrico Motta, Using the semantic web as background knowledge for ontology mapping, in *Ontology Matching* (2006).

35. Rion Snow, Daniel Jurafsky and Andrew Y. Ng, Learning syntactic patterns for automatic hypernym discovery, in *Advances in Neural Information Processing Systems* (*NIPS*) (2004).
36. Michael Strube and Simone Paolo Ponzetto, Wikirelate! Computing semantic relatedness using Wikipedia, in *Proc. 21st National Conf. on Artificial Intelligence* (AAAI Press, 2006), pp. 1419–1424.
37. Fabian M. Suchanek, Gjergji Kasneci and Gerhard Weikum, YAGO: A core of semantic knowledge, in *Proc. 16th ACM Conf. on World Wide Web* (*WWW*) (2007), pp. 697–706.
38. Asuka Sumida and Kentaro Torisawa, Hacking Wikipedia for hyponymy relation acquisition, in *Proc. of IJCNLP 2008* (2008), pp. 883–888.