1

Ontology Learning

Alexander Maedche[1] and Steffen Staab[2]

[1] FZI Research Center for Information Technologies, University of Karlsruhe, Germany
email: maedche@fzi.de
[2] Institute AIFB, University of Karlsruhe, Germany
email: sst@aifb.uni-karlsruhe.de

Summary. *Ontology Learning* greatly facilitates the construction of ontologies by the ontology engineer. The notion of ontology learning that we propose here includes a number of complementary disciplines that feed on different types of unstructured and semi-structured data in order to support a semi-automatic, cooperative ontology engineering process. Our ontology learning framework proceeds through ontology import, extraction, pruning, and refinement, giving the ontology engineer a wealth of coordinated tools for ontology modelling. Besides of the general architecture, we show in this paper some exemplary techniques in the ontology learning cycle that we have implemented in our ontology learning environment, KAON Text-To-Onto.

1.1 Introduction

Ontologies constitute a formal conceptualization of a particular domain of interest that is shared by a group of people. When building ontologies into information systems, it is possible to modularize many software aspects mostly related to the domain (e.g., taxonomic structures) from ones mostly related to the processing (e.g., querying) and visualization (e.g., layouting) of data.

One could argue that the drawback one encounters there is that such information systems software cannot be built with an implicit understanding of the domain, but rather it is necessary to make conceptualizations of the domain explicit — which may be a difficult task, resulting in a well-known knowledge engineering bottleneck. While one answer to this argument, also found in software engineering, certainly is: you should make your structures explicit in order to be able to adapt and extend them easily, the quest for faster and cheaper ontology engineering remains. Though ontology engineering tools have matured over the last decade, the manual building of ontologies still remains a tedious, cumbersome task.

Thus, when using ontologies as a basis for information systems, one has to face questions about development time, difficulty, confidence and the maintenance of ontologies. Thus, what one ends up with is similar to what knowledge engineers have dealt with over the last two decades when elaborating methodologies for knowledge acquisition or workbenches for defining knowledge bases. A method which has proven to be extremely beneficial for the knowledge acquisition task is the integration of knowledge acquisition with machine learning techniques.

Ontology Learning [22] aims at the integration of a multitude of disciplines in order to facilitate the construction of ontologies, in particular ontology engineering [48, 11] and machine learning. Because the fully automatic acquisition of knowledge by machines remains in the distant future, the overall process is considered to be semi-automatic with human intervention. It relies on the "balanced cooperative modeling" paradigm [32], describing a coordinated interaction between human modeler and learning algorithm for the construction of ontologies. This objective in mind, an approach that combines ontology engineering with machine learning is described here.

Organization

This chapter is organized as following. Section 1.2 introduces a generic architecture for ontology learning and its relevant components. In Section 1.3 we introduce various complementary basic ontology learning algorithms that may serve as a basis for ontology learning. Section 1.4 describes how we have implemented our notion of ontology learning in the form of a concrete system called KAON Text-To-Onto. Section 1.5 surveys related work.

1.2 An Architecture and Process Model for Ontology Learning

The purpose of this section is to introduce a generic ontology learning architecture and its four major components, before we continue in detail describing the conceptual model we have developed for our KAON Text-To-Onto system.

Thereby, our process model builds on the principal idea of data mining as a process (e.g., [6]) with the phases of business and data understanding, data preparation, modeling, evaluation and deployment. This implies that our notion of ontology learning makes all process steps transparent — in contrast to some focused ontology learning application for which one may decide to configure a concrete processing pipeline. The latter would have to be configured from the general modules provided in KAON Text-To-Onto.

- **Ontology Management Component**: The ontology engineer uses the ontology management component to manually deal with ontologies. In particular, it allows for the inclusion of existing ontologies, their browsing, validation [47], modification, versioning, and evolution [28].
- **Resource Processing Component**: This component contains a wide range of techniques for *discovering, importing, analyzing and transforming* relevant input data. An important sub-component is a natural language processing system. The general task of the resource processing component is to generate a set of pre-processed data as input for the algorithm library component.

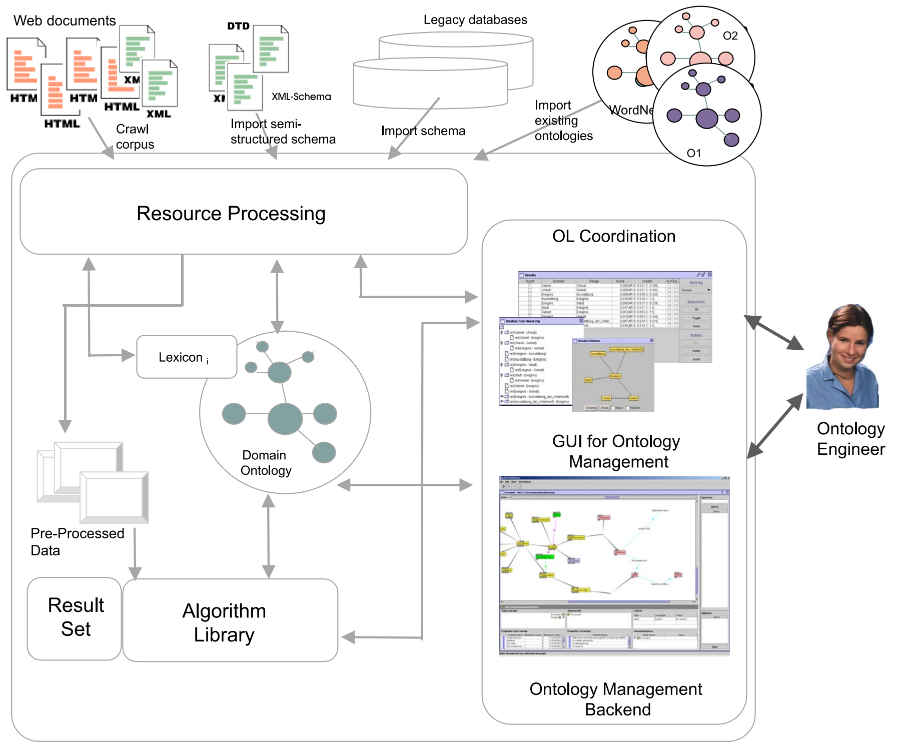


Fig. 1.1. Ontology Learning Conceptual Architecture

- **Algorithm Library Component**: This component acts as the algorithmic backbone of the framework. A number of algorithms are provided for the extraction and maintenance of the ontology parts contained in the ontology model. In order to be able to combine the extraction results of different learning algorithms, it is necessary to standardize the output in a common way. Therefore a common result structure for all learning methods is provided. If several extraction algorithms obtain overlapping or complementary results, they are combined and presented to the user only once.
- **Coordination Component**: The ontology engineer uses this component to interact with the ontology learning components for resource processing and for the algorithm library. Comprehensive user interfaces are provided to the ontology engineer to help select relevant data, apply processing and transformation techniques or start a specific extraction mechanism. Data processing can also be triggered by the selection of an ontology learning algorithm that requires a specific representation. Results are merged using the result set structure and presented to the ontology engineer with different views of the ontology structures.

In the following we provide a detailed overview of the four components.

1.2.1 Ontology Management

As core to our approach we have built on our ontology management and application infrastructure called KArlsruhe ONtology and Semantic Web Infrastructure (KAON; also cf. [38]), allowing easy ontology management and application.

KAON is based on an ontology model as introduced in [36]. Briefly, the ontology language is based on RDF(S), but with clean separation of modeling primitives from the ontology itself (thus avoiding the pitfalls of self-describing RDFS primitives such as subClassOf), providing means for modelling meta-classes and incorporating several commonly used modelling primitives, such as transitive, symmetric and inverse properties, or cardinalities. All information is organized in so-called OI-models (ontology-instance models), containing both ontology entities (concepts and properties) as well as their instances. This allows the grouping of concepts with their well-known instances into self-contained units. E.g. a geographical information OI-model might contain the concept Continent along with its seven well-known instances. An OI-model may include another OI-model, thus making all definitions from the included OI-model automatically available (cf. [23]).

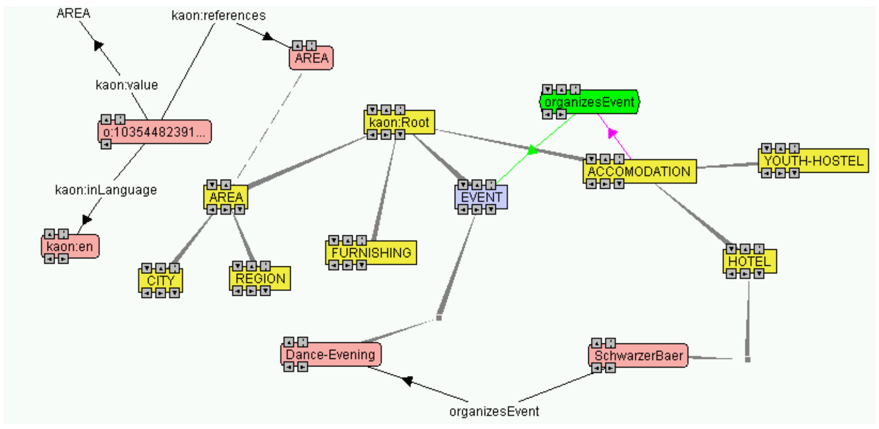


Fig. 1.2. OI-model example

A lexical OI-model extends an OI-model by specific entries that reflect various lexical properties of ontology entities. For instance, concepts like PLANET-VENUS (but also relations) have a label (e.g., ‘Venus’), synonyms (e.g., ‘evening star’, ‘morning star’), a lexical stem and textual documentation.

There is an $n : m$ relationship between lexical entries and instances, established by the property REFERENCES. Thus, the same lexical entry (e.g., ‘his jaguar’) may be associated with several elements (e.g., an instance representing a Jaguar car or a jaguar cat). The value, i.e. its literal occurrence, of the lexical entry is given by the property VALUE, whereas the language of the value is specified through the INLANGUAGE property that refers to the set of all languages as defined by the ISO standard 639.

1.2.2 Coordination Component

The ontology engineer uses the coordination component to select input data, i.e. relevant resources such as HTML documents from the Web that are exploited in the further discovery process. Secondly, using the coordination component, the ontology engineer also chooses among a set of resource processing methods available at the resource processing component and among a set of algorithms available in the algorithm library.

1.2.3 Resource processing

As mentioned earlier this component contains a wide range of techniques for *discovering, importing, analyzing and transforming* relevant input data. An important sub-component is a natural language processing system. The general task of the resource processing component is to generate a set of pre-processed data as input for the algorithm library component.

Resource processing strategies differ depending on the type of input data made available:

- Semi-structured documents, like dictionaries, may be transformed into a predefined relational structure as described in [16]. HTML documents may be indexed and reduced to free text.
- For processing free text the system accesses language dependent natural language processing systems. E.g., for German we have used SMES (Saarbrücken Message Extraction System), a shallow text processor [34]. SMES comprises a *tokenizer* based on regular expressions, a *lexical analysis* component including various word *lexicons*, a *morphological analysis* module, a *named entity recognizer*, a *part-of-speech tagger* and a *chunk parser*. For English, one may build on many available language processing resource, e.g. the GATE system as described in [7]. In the currently supported implementation that is described in Section 1.4 we just use the well-known Porter stemming algorithm [41] as a very simple means for normalizing natural language terms.

After preprocessing according to one of these or similar strategies, the resource processing module transforms given data into an algorithm-specific relational representation.

1.2.4 Algorithm Library

An ontology may be described by sets of concepts, relations, lexical entries, and links between these entities. An ontology may be built following this specification using various algorithms working on the preprocessed input data. Thereby, different algorithms may generate definitions for overlapping as well as complementary ontology parts. For instance, an information extraction module specialized for processing appositions[3] may find subclass relationships similar to classification with statistical techniques by kNN (cf., e.g., [39]). Machine learning with association rules, however, may rather be used to detect general binary relationships [24]. Hence, we may reuse algorithms from the library for acquiring different parts of the ontology definition.

Subsequently, we introduce some of the algorithms available in our implementation. In general, we use a result combination approach, i.e. each algorithm that is provided by the library generates normalized results that follow the ontology structures sketched above and contributes to a coherent ontology definition. In the future, more research is needed on how overlapping results may contribute to a common result in a really multi-strategy learning fashion.

1.3 Ontology Learning Algorithms

Some example learning algorithms are described here. They cover different parts of the ontology definition — parts that may also be evaluated in isolation of each other [26].

1.3.1 Lexical Entry & Concept Extraction

A simple technique for extracting relevant lexical entries that may indicate concepts is counting frequencies of terms in a given set of (linguistically preprocessed) documents, the corpus $\mathcal{D}$. In general this approach is based on the assumption that a frequent term in a set of domain-specific texts indicates occurrence of a relevant concept. Research in information retrieval has shown that there are more effective methods of term weighting than simple counting of frequencies. A standard information retrieval approach is pursued for term weighting, based on the following measures.

- The **lexical entry frequency** $\mathbf{lef}_{l,d}$ is the frequency of occurrence of lexical entry $l \in \mathcal{L}$ in a document $d \in \mathcal{D}$.
- The **document frequency** $\mathbf{df}_l$ is the number of documents in the corpus $\mathcal{D}$ that l occurs in.
- The **corpus frequency** $\mathbf{cf}_l$ is total number of occurrences of l in the overall corpus $\mathcal{D}$.

The reader may note that $\mathrm{df}_l \leq \mathrm{cf}_l$ and $\sum_d \mathrm{lef}_{l,d} = \mathrm{cf}_l$. The relevance of terms is measured based on the information retrieval measure **tfidf (term frequency inverted document frequency)**.

Definition 1. *Let $lef_{d,l}$ be the term frequency of the lexical entry l in a document d. Let df_l be the overall document frequency of lexical entry l. Then $tfidf_{l,d}$ of the lexical entry l for the document d is given by:*

[3] An example for an apposition is given in the following sentence. "Ryokans, a typical Japanese accomodation, are frequented by Europeans, too". There accomodation is an apposition to and a superconcept of Ryokan.

$$tfidf_{l,d} = lef_{l,d} * \log\left(\frac{|\mathcal{D}|}{df_l}\right). \tag{1.1}$$

Tfidf weighs the frequency of a lexical entry in a document with a factor that discounts its importance when it appears in almost all documents. Therefore terms that appear too rarely or too frequently are ranked lower than terms that hold the balance. To rank the importance of a term not only for a document, but for a whole corpus, tfidf values for lexical entries l are computed as follows:[4]

Definition 2. $$tfidf_l := \sum_{d \in \mathcal{D}} tfidf_{l,d}, \qquad tfidf_l \in \boldsymbol{R}. \tag{1.2}$$

The user may define and vary a **threshold** $k \in \boldsymbol{R}^+$ that tfidf$_l$ has to exceed. This threshold is then used to explore terms from the corpus for inclusion into the set of lexical entries and possibly into the concept hierarchy.

1.3.2 Extraction of Taxonomic Relations

The extraction of taxonomic relations may be done in various ways. In our framework we mainly include the following three kinds of approaches:

- Statistics-based extraction using clustering
- Statistics-based extraction using classification
- Lexico-syntactic pattern extraction

In the following we will shortly elaborate on the three approaches.

Clustering

Distributional data about words may be used to build concepts and their embedding into a hierarchy "from scratch" using clustering mechanisms.

A distributional representation describes a term by the (weighted) frequency of terms that occur in a delineated context. The delineation may be defined by sequential vicinity of terms ("How many terms appear between the represented and the representing term?" — which is what we use), by a syntactic delineation ("Which terms may have a syntactic dependency to the represented term?") or by some text structural criterion ("What terms appear in the same paragraph delineated by HTML tags?").

Clustering can be defined as the process of organizing objects into groups whose members are similar in some way based on the distributional representation (see [15]). In general there are three major styles of clustering:

1. Agglomerative: In the initialization phase, each term is defined to constitute a cluster of its own. In the growing phase larger clusters are iteratively generated by merging the most similar/least dissimilar ones until some stopping criterion is reached.

[4] A list of stopwords being excluded.

2. Partitional: In the initialization phase, the set of all terms is a cluster. In the refinement phase smaller clusters are (iteratively) generated by splitting the largest cluster or the least homogeneous cluster into several subclusters.
   Both, agglomerative and partitional clustering techniques, are used to produce hierarchical descriptions of terms. Both rely on notions of (dis-)similarity, for which a range of measures exist (e.g., Jacquard, Kullback-Leibler divergence, L1-norm, cosine; cf. [21]).
   In practice, partitional clustering (like the K-Means clustering technique) is faster as it can be performed in runtime of $\mathcal{O}(n)$ compared to $\mathcal{O}(n^2)$ for agglomerative techniques, where n is the number of represented terms.
3. Conceptual: Conceptual clustering builds a lattice of terms by investigating the exact overlap of representing terms between two represented terms. In the worst case, the complexity of the resulting concept lattice is exponential in n. Thus, people either just compute a sublattice [46] or use a heuristics.

Either way one may construct a hierarchy of term clusters for detailed inspection by the ontology engineer.

Classification

When a substantial hierarchy is already given, e.g. by basic level categories from a general resource like WordNet [31], one may rather decide to refine the taxonomy by classifying new relevant terms into the given concept hierarchy. The distributional representation described above is then used to learn a classifier from a training corpus and the set of predefined concepts with their lexical entries.

Afterwards, one may construct the distributional representations of relevant, unclassified terms and let the learned classifier propose a node on which to subclass the new term (cf, e.g., [39]). k nearest neighbor (kNN) and support vector machines are typical learning algorithms exploited for this purpose.

Lexico-Syntactic Patterns

The idea of using lexico-syntactic patterns in the form of regular expressions for the extraction of semantic relations, in particular taxonomic relationships, has been introduced by [16]. Pattern-based approaches in general are heuristic methods using regular expressions that originally have been successfully applied in the area of information extraction. In this lexico-syntactic ontology learning approach the text is scanned for instances of distinguished lexico-syntactic patterns that indicate a relation of interest, e.g. the taxonomy. Thus, the underlying idea is very simple: Define a regular expression that captures re-occurring expressions and map the results of the matching expression to a semantic structure, such as taxonomic relations between concepts.

Example

In [16] the following lexico-syntactic pattern is considered

$$\ldots NP\{, NP\} * \{,\} \text{ or other } NP \ldots$$

When we apply this pattern to a sentence it can be infered that the NP's referring to concepts on the left of *or other* are sub concepts of the NP referring to a concept on the right. For example from the sentence

Bruises, wounds, broken bones or other injuries are common.

we extract the taxonomic relations (Bruise,Injury), (Wound,Injury) and (Broken-Bone,Injury).

Within our ontology learning system we have applied different techniques to dictionary definitions in the context of the insurance and telecommunication domains as described in [19]. An important aspect in this system and approach is that existing concepts are included in the overall process. In contrast to [16] the extraction operations have been performed on the concept level. Thus, patterns have been directly matched onto concepts. Hence, besides of extracting taxonomic relations from scratch, the system can refine existing relationships and refer to existing concepts.

1.3.3 Extraction of General Binary Relationships

Association rules have been established in the area of data mining, thus, finding interesting association relationships among a large set of data items. Many companies become interested in mining association rules from their databases (e.g. for helping in many business decisions such as customer relationship management, cross-marketing and loss-leader analysis. A typical example of association rule mining is market basket analysis. This process analyzes customer buying habits by finding associations between the different items that customers place in their shopping baskets. The information discovered by association rules may help to develop marketing strategies, e.g. layout optimization in supermarkets (placing milk and bread within close proximity may further encourage the sale of these items together within single visits to the store). In [1] concrete examples for extracted associations between items are given. The examples are based on supermarket products that are included in a set of transactions collected from customers' purchases. One classical anectode is that "diapers are purchased together with beer".

For the objective of ontology learning, this data mining algorithm may be applied to syntactic structures and statistical co-occurrences appearing in text. To illustrate its working, we here show an example. The example is based on actual ontology learning experiments as described in [24]. A text corpus given by a WWW provider for tourist information has been processed. The corpus describes actual objects referring to locations, accomodations, furnishings of accomodations, administrative information, or cultural events, such as given in the following example sentences.

(1) a. "Mecklenburg's" schönstes "Hotel" liegt in Warnemuende. ("Mecklenburg's" most beautiful "hotel" is located in Warnemuende.)
b. Ein besonderer Service für unsere Gäste ist der "Fris örsalon" in unserem "Hotel". (A "hairdresser" in our "hotel" is a special service for our guests.)

c. Das Hotel Mercure hat "Balkone" mit direktem "Strandzugang". (The hotel Mercure offers "balconies" with direct "access" to the beach.)
d. Alle "Zimmer" sind mit "TV", Telefon, Modem und Minibar ausgestattet. (All "rooms" have "TV", telephone, modem and minibar.)

Processing the example sentences (1a) and (1b) the dependency relations between the terms are extracted (and some more). In sentences (1c) and (1d) the heuristic for prepositional phrase-attachment (minimal attachment: attach a prepositional phrase to its nearest noun phrase) and the sentence heuristic[5] relate pairs of lexical entries, respectively. Thus, four concept pairs – among many others – are derived with knowledge from the lexicon.

Table 1.1. Examples for Linguistically Related Pairs of Concepts

L_1	$a_{i,1}$	L_2	$a_{i,2}$
"Mecklenburgs"	AREA	*hotel*	HOTEL
"hairdresser"	HAIRDRESSER	*hotel*	HOTEL
"balconies"	BALCONY	*access*	ACCESS
"room"	ROOM	*TV*	TELEVISION

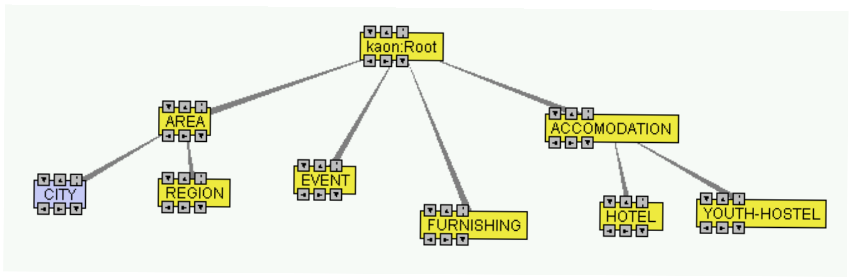


Fig. 1.3. An Example Concept Taxonomy as Background Knowledge for Extracting General Binary Relationships

The algorithm for learning generalized association rules uses a taxonomy. The taxonomy may be built with methods such as described in Section 1.3.2. An excerpt involving the concept pairs from above (among some more) is depicted in Figure 1.3. In our actual experiments, the algorithm discovered a large number of interesting and important general binary conceptual relationships. A few of them are listed in Table 1.2. Note that in this table we also list two conceptual pairs, viz. (AREA, HOTEL) and (ROOM, TELEVISION), that are not presented to the user, but that are pruned. The

[5] The sentence heuristic essentially says: if no parse tree can be built, relate concepts represented by nouns that are adjacent to each other – abstracting from terms that do not refer to concepts or relationships.

reason is that there are ancestral association rules, viz. (AREA, ACCOMODATION) and (ROOM,FURNISHING), respectively with higher confidence and support measures.

Table 1.2. Examples of Discovered General Binary Relationships

Discovered relation	Confidence	Support
(AREA, ACCOMODATION)	0.38	0.04
~~(AREA, HOTEL)~~	~~0.1~~	~~0.03~~
(ROOM, FURNISHING)	0.39	0.03
~~(ROOM, TELEVISION)~~	~~0.29~~	~~0.02~~
(ACCOMODATION, ADDRESS)	0.34	0.05
(RESTAURANT, ACCOMODATION)	0.33	0.02

Other algorithms use verbs as potential indicators for general binary relationships and acquire selectional restrictions from corpus data [5].

1.3.4 Ontology Pruning

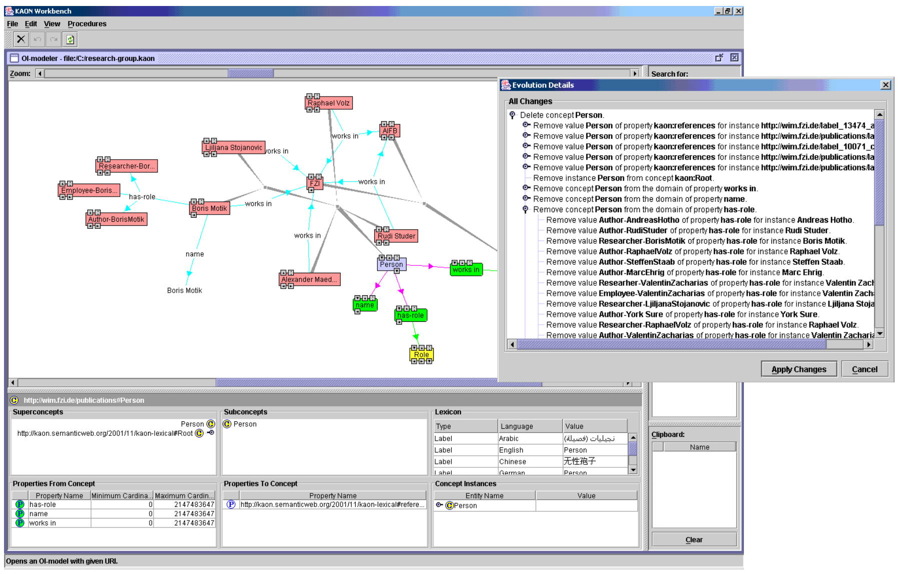
Pruning is needed, if one adopts a (generic) ontology to a given domain. We assume that the occurrence of specific concepts and conceptual relations in web documents are vital for the decision whether or not a given concept or relation should remain in an ontology. We exploit a frequency-based approach determining concept frequencies in a corpus. Entities that are frequent in a given corpus are considered as a constituent of a given domain. But - in contrast to ontology extraction - the mere frequency of ontological entities is not sufficient.

To determine domain relevance, ontological entities retrieved from a domain corpus are compared to frequencies obtained from a generic corpus. The user can select several relevance measures for frequency computation. The ontology pruning algorithm uses the computed frequencies to determine the relative relevancy of each concept contained in the ontology. All existing concepts and relations, which are more frequent in the domain-specific corpus remain in the ontology. The user can also control the pruning of concepts which are neither contained in the domain-specific nor in the generic corpus.

1.4 KAON Text-To-Onto

This section describes the implemented ontology learning system KAON Text-To-Onto that is embedded in KAON, the Karlsruhe Ontology and Semantic web infrastructure (cf. [38]). KAON[6] is an open-source ontology management and application infrastructure targeted for semantics-driven business applications. It includes a comprehensive tool suite allowing easy ontology management and application.

[6] http://kaon.semanticweb.org



Evolution Details

All Changes

- Delete concept **Person**.
  - Remove value **Person** of property **kaon:references** for instance **http://wim.fzi.de/label_13474_a**
  - Remove value **Person** of property **kaon:references** for instance **http://wim.fzi.de/publications/la**
  - Remove value **Person** of property **kaon:references** for instance **http://wim.fzi.de/label_10071_c**
  - Remove value **Person** of property **kaon:references** for instance **http://wim.fzi.de/publications/la**
  - Remove value **Person** of property **kaon:references** for instance **http://wim.fzi.de/publications/la**
  - Remove instance **Person** from concept **kaon:Root**.
  - Remove concept **Person** from the domain of property **works in**.
  - Remove concept **Person** from the domain of property **name**.
  - Remove concept **Person** from the domain of property **has-role**.
    - Remove value **Author-AndreasHotho** of property **has-role** for instance **Andreas Hotho**.
    - Remove value **Author-RudiStuder** of property **has-role** for instance **Rudi Studer**.
    - Remove value **Researcher-BorisMotik** of property **has-role** for instance **Boris Motik**.
    - Remove value **Author-RaphaelVolz** of property **has-role** for instance **Raphael Volz**.
    - Remove value **Author-SteffenStaab** of property **has-role** for instance **Steffen Staab**.
    - Remove value **Author-MarcEhrig** of property **has-role** for instance **Marc Ehrig**.
    - Remove value **Researher-ValentinZacharias** of property **has-role** for instance **Valentin Zach**
    - Remove value **Employee-ValentinZacharias** of property **has-role** for instance **Valentin Zach**
    - Remove value **Researcher-LjiljanaStojanovic** of property **has-role** for instance **Ljiljana Stoja**
    - Remove value **Author-York Sure** of property **has-role** for instance **York Sure**.
    - Remove value **Researcher-RaphaelVolz** of property **has-role** for instance **Raphael Volz**.
    - Remove value **Author-ValentinZacharias** of property **has-role** for instance **Valentin Zacharia**

Apply Changes Cancel

http://wim.fzi.de/publications#Person

Superconcepts

Person
http://kaon.semanticweb.org/2001/11/kaon-lexical#Root

Subconcepts

Person

Lexicon

Type	Language	Value
Label	Arabic	نجيليات (فصيلة)
Label	English	Person
Label	Chinese	无性孢子
Label	German	Person

Properties From Concept

Property Name	Minimum Cardina...	Maximum Cardin...
has-role	0	2147483647
name	0	2147483647
works in	0	2147483647

Properties To Concept

Property Name
http://kaon.semanticweb.org/2001/11/kaon-lexical#refere...

Concept Instances

Entity Name	Value
Person	

Fig. 1.4. KAON OI-Modeler

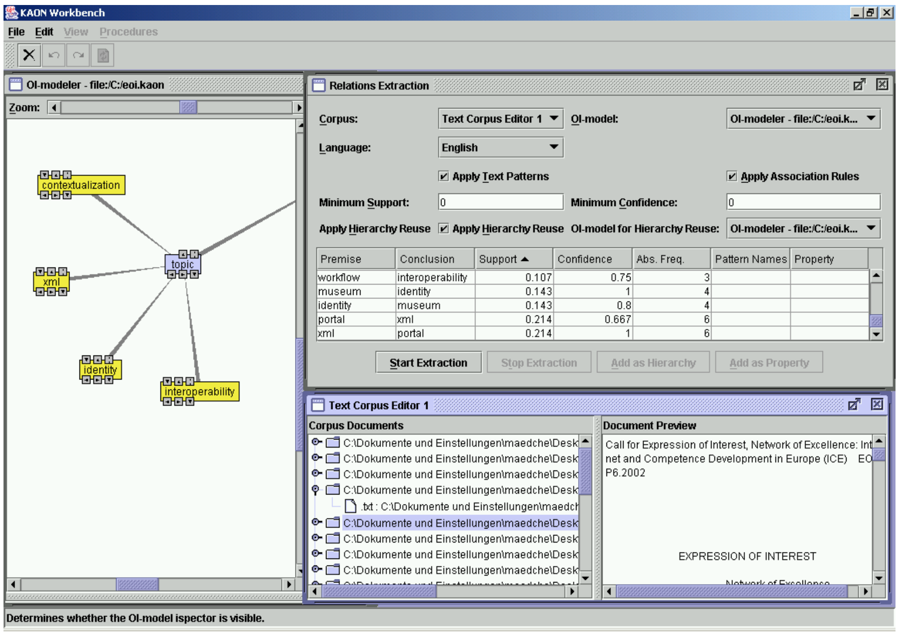
Within KAON we have developed KAON OI-modeler, an end-user application that realizes a graph-based user interface for OI-model creation and evolution [28]. Figure 1.4 shows the graph-based view onto an OI-model.

A specific aspect of KAON OI-modeler is that it supports ontology evolution at the user level (see right side of the screenshot). The figure shows a modelling session where the user attempted to remove the concept Person. Before applying this change to the ontology, the system computed the set of additional changes that must be applied. The tree of dependent changes is presented to the user, thus allowing the user to comprehend the effects of the change before it is actually applied. Only when the user agrees, the changed will be applied to the ontology.

KAON Text-To-Onto[7] builds on the KAON OI-modeler. KAON Text-To-Onto provides means for defining a corpus as a basis for ontology learning (see right lower part of Figure 1.5). On top of the corpus various algorithms may be applied. In the example depicts in Figure 1.5 the user has selected different parameters and executed a pattern-based extraction and an association rule extraction algorithm in parallel. KAON Text-To-Onto implements the aforementioned algorithms, though not (yet!) partitional and conceptual clustering.

Finally, the results are presented to the user (see Figure 1.5) and graphical means for adding lexical entries, concepts and conceptual relations are provided.

[7] KAON Text-To-Onto is open-source and available for download at the KAON Web page.



Corpus: Text Corpus Editor 1
OI-model: OI-modeler - file:/C:/eoi.k...
Language: English
☑ Apply Text Patterns
☑ Apply Association Rules
Minimum Support: 0
Minimum Confidence: 0
Apply Hierarchy Reuse ☑ Apply Hierarchy Reuse
OI-model for Hierarchy Reuse: OI-modeler - file:/C:/eoi.k...

Premise	Conclusion	Support ▲	Confidence	Abs. Freq.	Pattern Names	Property
workflow	interoperability	0.107	0.75	3		
museum	identity	0.143	1	4		
identity	museum	0.143	0.8	4		
portal	xml	0.214	0.667	6		
xml	portal	0.214	1	6		

```
C:\Dokumente und Einstellungen\maedche\Desk
C:\Dokumente und Einstellungen\maedche\Desk
C:\Dokumente und Einstellungen\maedche\Desk
C:\Dokumente und Einstellungen\maedche\Desk
    .txt : C:\Dokumente und Einstellungen\maedch
C:\Dokumente und Einstellungen\maedche\Desk
C:\Dokumente und Einstellungen\maedche\Desk
C:\Dokumente und Einstellungen\maedche\Desk
C:\Dokumente und Einstellungen\maedche\Desk
```

Call for Expression of Interest, Network of Excellence: Int
net and Competence Development in Europe (ICE) EO
P6.2002

EXPRESSION OF INTEREST

Fig. 1.5. KAON Text-To-Onto

1.5 Related Work

Until recently ontology learning *per se*, i.e. for comprehensive construction of ontologies, has not existed. We here give the reader a comprehensive overview of existing work that has actually researched and practiced techniques for solving parts of the overall problem of ontology learning.

There are only a few approaches that described the development of frameworks and workbenches for extracting ontologies from data: Faure & Nedellec [9] present a cooperative machine learning system, ASIUM, which acquires taxonomic relations and subcategorization frames of verbs based on syntactic input. The ASIUM system hierarchically clusters nouns based on the verbs that they are syntactically related with and *vice versa*. Thus, they cooperatively extend the lexicon, the set of concepts, and the concept heterarchy. A more recent approach is described by Missikoff et al. [35] who combine an ontology engineering environment with focus on consensus building with sophisticated tools for extracting concepts that have multi-word lables (e.g. ‘swimming pool’) and for extracting relationships other than taxonomy, e.g. pertainance.

Hahn and Schnattinger [14] introduced a methodology for the maintenance of domain-specific taxonomies. An ontology is incrementally updated as new concepts are acquired from real-world texts. The acquisition process is centered around linguistic and conceptual “quality” of various forms of evidence underlying the generation and refinement of concept hypotheses. Their ontology learning approach is embedded in a framework for natural language understanding, named Syndicate [13], and they have recently extended their framework to a dual learner for grammars and ontologies [12].

Mikheev & Finch [30] have presented their KAWB Workbench for "Acquisition of Domain Knowledge form Natural Language". The workbench comprises a set of tools for uncovering internal structure in natural language texts. The main idea behind the workbench is the independence of the text representation and text analysis phases. At the representation phase the text is converted from a sequence of characters to features of interest by means of the annotation tools. At the analysis phase those features are used by statistics gathering and inference tools for finding significant correlations in the texts. The analysis tools are independent of particular assumptions about the nature of the feature-set and work on the abstract level of feature elements represented as SGML items.

Much work in a number of disciplines — computational linguistics, information retrieval, machine learning, databases, software engineering — has actually researched and practiced techniques for solving part of the overall problem. Hence, techniques and methods relevant for ontology learning may be found under terms like the acquisition of selectional restrictions (cf. Resnik [42] and Basili et al. [4]), word sense disambiguation and learning of word senses (cf. Hastings [50]), the computation of concept lattices from formal contexts (cf. Ganter & Wille [10]) and Reverse Engineering in software engineering (cf. Mueller et al. [37]).

Ontology Learning puts a number of research activities, which focus on different types of inputs, but share their target of a common domain conceptualization, into one perspective. One may recognize that these activities are spread between very different communities. Further references may in particular be found at the three ontology learning workshops [44, 27, 3].

1.6 Conclusion

We have introduced ontology learning as an approach that may greatly facilitate the construction of ontologies by the ontology engineer. The notion of Ontology Learning introduced in this article aims at the integration of a multitude of disciplines in order to facilitate the construction of ontologies. The overall process is considered to be semi-automatic with human intervention. It relies on the "balanced cooperative modeling" paradigm, describing a coordinated interaction between human modeler and learning algorithm for the construction of ontologies for the Semantic Web.

References

1. Agrawal, R. and Imielinski, T. and Swami, A.: Mining Associations between Sets of Items in Massive Databases, In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993.

2. H. Assadi. Construction of a regional ontology from text and its use within a documentary system. In *Proceedings of the International Conference on Formal Ontology and Information Systems - FOIS'98*, Trento, Italy, 1998.
3. N. Aussenac-Gilles, and A. Maedche (eds.). *Workshop on Machine Learning and Natural Language Processing for Ontology Engineering*, http://www-sop.inria.fr/acacia/OLT2002
4. R. Basili, M. T. Pazienza, and P. Velardi. Acquisition of selectional patterns in a sublanguage. *Machine Translation*, 8(1):175–201, 1993.
5. Paul Buitelaar. CoreLex*: Systematic Polysemy and Underspecification*. PhD thesis, Brandeis University, Department of Computer Science, 1998.
6. P. Chapman, R. Kerber, J. Clinton, T. Khabaza, T. Reinartz, and R. Wirth. The CRISP-DM Process Model. Discussion Paper, March 1999. http://www.crisp-dm.org/
7. H. Cunningham and R. Gaizauskas and K. Humphreys and Y. Wilks: Three Years of GATE, In Proceedings of the AISB'99 Workshop on Reference Architectures and Data Standards for NLP, Edinburgh, U.K. Apr, 1999.
8. F. Esposito, S. Ferilli, N. Fanizzi, and G. Semeraro. Learning from parsed sentences with inthelex. In *Proceedings of Learning Language in Logic Workshop (LLL-2000), Lisbon, Portugal, 2000*, 2000.
9. D. Faure and C. Nedellec. A corpus-based conceptual clustering method for verb frames and ontology acquisition. In *LREC workshop on adapting lexical and corpus resources to sublanguages and applications*, Granada, Spain, 1998.
10. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin - Heidelberg - New York, 1999.
11. A. Gomez-Perez. Ontology Engineering. Springer Verlag, 2002/2003.
12. U. Hahn and K. Marko: An integrated, dual learner for grammars and ontologies. *Data and Knowledge Engineering*, 42(3): 273-291, 2002.
13. U. Hahn and M. Romacker. Content management in the syndikate system — how technical documents are automatically transformed to text knowledge bases. *Data & Knowledge Engineering*, 35:137–159, 2000.
14. U. Hahn and K. Schnattinger. Towards text knowledge engineering. In *Proc. of AAAI '98*, pages 129–144, 1998.
15. L. Kaufman and P. Rousseeuw: Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley, 1990.
16. Hearst, M.: Automatic acquisition of hyponyms from large text corpora. In Proceedings of the 14th International Conference on Computational Linguistics. Nantes, France, 1992.
17. J. Jannink and G. Wiederhold. Thesaurus entry extraction from an on-line dictionary. In *Proceedings of Fusion '99, Sunnyvale CA, July 1999*, 1999. http://www-db.stanford.edu/SKC/publications.html.
18. P. Johannesson. A method for transforming relational schemas into conceptual schemas. In M. Rusinkiewicz, editor, *10th International Conference on Data Engineering*, pages 115 – 122, Houston, 1994. IEEE Press.
19. Kietz, J.-U. and Volz, R. and Maedche, A.: Semi-automatic ontology acquisition from a corporate intranet. In International Conference on Grammar Inference (ICGI-2000), Lecture Notes in Artificial Intelligence, LNAI, 2000.
20. J.-U. Kietz and K. Morik. A polynomial approach to the constructive induction of structural knowledge. *Machine Learning*, 14(2):193–218, 1994.
21. L. Lee. Measures of distributional similarity. In *Proc. of the 37th Annual Meeting of the Association for Computational Linguistics*, 1999, pp. 25-32.
22. A. Maedche: Ontology Learning for the Semantic Web. Kluwer Academic Publishers, 2002.

23. A. Maedche, B. Motik, L. Stojanovic, and R. Studer. ??title?? In *Proceedings 12th International World Wide Web Conference (WWW12),* Semantic Web Track, 2003, Budapest, Hungary.
24. A. Maedche and S. Staab. Discovering conceptual relations from text. In *Proceedings of ECAI-2000*. IOS Press, Amsterdam, 2000.
25. A. Maedche and S. Staab. Mining ontologies from text. In Proceedings of EKAW-2000, Springer Lecture Notes in Artificial Intelligence (LNAI-1937), Juan-Les-Pins, France, 2000.
26. A. Maedche and S. Staab. Measuring Similarity between Ontologies. In: *Proc. Of the European Conference on Knowledge Acquisition and Management - EKAW-2002*. Madrid, Spain, October 1-4, 2002. LNCS/LNAI 2473, Springer, 2002, pp. 251-263.
27. A. Maedche, S. Staab, E. Hovy, and C. Nedellec (eds.). *The IJCAI-2001 Workshop on Ontology Learning. Proceedings of the Second Workshop on Ontology Learning - OL'2001*, Seattle, WA, USA, August 4, 2001. CEUR Proceedings.
28. A. Maedche, B. Motik, L. Stojanovic, R. Studer, and R. Volz: Ontologies for Enterprise Knowledge Management, IEEE Intelligent Systems, December, 2002.
29. Manning, C. and Schuetze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge, Massachusetts, 1999.
30. A. Mikheev and S. Finch. A workbench for finding structure in text. In *In Proceedings of the 5th Conference on Applied Natural Language Processing — ANLP'97, March 1997, Washington DC, USA*, pages 372–379, 1997.
31. G. Miller. WordNet: A Lexical Database for English. *Communications of the ACM,* 38(11), pp. 3941.
32. K. Morik and S. Wrobel and J.-U. Kietz and W. Emde *Knowledge acquisition and machine learning: Theory, methods, and applications,* London: Academic Press, 1993.
33. E. Morin. Automatic acquisition of semantic relations between terms from technical corpora. In *Proc. of the Fifth International Congress on Terminology and Knowledge Engineering - TKE'99*, 1999.
34. G. Neumann and R. Backofen and J. Baur and M. Becker and C. Braun: An Information Extraction Core System for Real World German Text Processing. In Proceedings of ANLP-97, Washington, USA, 1997.
35. M. Missikoff, R. Navigli, and P. Velardi: The Usable Ontology: An Environment for Building and Assessing a Domain Ontology. *Proceedings of the International Semantic Web Conference 2002*. Springer, 2002, pp. 39-53.
36. B. Motik and A. Maedche and R. Volz: A Conceptual Modeling Approach for building semantics-driven enterprise applications. 1st International Conference on Ontologies, Databases and Application of Semantics (ODBASE-2002), California, USA, 2002.
37. H. A. Mueller, J. H. Jahnke, D. B. Smith, M.-A. Storey, S. R. Tilley, and K. Wong. Reverse Engineering: A Roadmap. In *Proceedings of the 22nd International Conference on Software Engineering (ICSE-2000), Limerick, Ireland*. Springer, 2000.
38. D. Oberle, R. Volz, S. Staab, and B. Motik. An extensible ontology software environment. In this book.
39. V. Pekar and S. Staab. Taxonomy Learning — Factoring the structure of a taxonomy into a semantic classification decision. In: *Proceedings of the 19th Conference on Computational Linguistics, COLING-2002*, August 24 – September 1, 2002, Taipei, Taiwan, 2002.
40. Pereira, F. and Tishby, N. and Lee, L.: Distributation Clustering of English Words. In Proceedings of the ACL-93, 1993.
41. Porter, M. F.: An algorithm for suffix stripping. In *Program*, 14(3), 1980, pp. 130137.

42. P. Resnik. *Selection and Information: A Class-based Approach to Lexical Relationships*. PhD thesis, University of Pennsylania, 1993.
43. S. Schlobach. Assertional mining in description logics. In *Proceedings of the 2000 International Workshop on Description Logics (DL2000)*, 2000. http://SunSITE.Informatik.RWTH-Aachen.DE/Publications/CEUR-WS/Vol-33/.
44. S. Staab, A. Maedche, C. Nedellec, and P. Wiemer-Hastings (eds.). *The ECAI'2000 Workshop on Ontology Learning. Proceedings of the First Workshop on Ontology Learning - OL'2000,* Berlin, Germany, August 25, 2000. CEUR Proceedings, Vol-31.
45. L. Stojanovic, N. Stojanovic, and R. Volz. A reverse engineering approach for migrating data-intensive web sites to the semantic web. In *IIP-2002, August 25-30, 2002, Montreal, Canada (Part of the IFIP World Computer Congress WCC2002),* 2002.
46. G. Stumme, R. Taouil, Y. Bastide, N. Pasqier, and L. Lakhal. Computing Iceberg Concept Lattices with Titanic. *Journal on Knowledge and Data Engineering,* 42(2), pp. 189222.
47. Y. Sure, J. Angele, and S. Staab. OntoEdit: Guiding Ontology Development by Methodology and Inferencing In S. Spaccapietra, S. March, and K. Aberer (eds.). LNCS - Semantics of Data, Springer, 2003 (to appear). (Extended version from ODBase-2002).
48. Y. Sure, S. Staab, and R. Studer. Methodology for Development and Employment of Ontology based Knowledge Management Applications In this book.
49. Z. Tari, O. Bukhres, J. Stokes, and S. Hammoudi. The Reengineering of Relational Databases based on Key and Data Correlations. In *Proceedings of the 7th Conference on Database Semantics (DS-7), 7-10 October 1997, Leysin, Switzerland*. Chapman & Hall, 1998.
50. P. Wiemer-Hastings, A. Graesser, and K. Wiemer-Hastings. Inferring the meaning of verbs from context. In *Proceedings of the Twentieth Annual Conference of the Cognitive Science Society*, 1998.
51. Y. Wilks, B. Slator, and L. Guthrie. *Electric Words: Dictionaries, Computers, and Meanings*. MIT Press, Cambridge, MA, 1996.

Table 1.3. Classification of Ontology Learning Approaches

Domain	Method	Features used	Prime purpose	Papers
Free Text	Clustering	Syntax	Extract	Buitelaar [5], Assadi [2] and Faure & Nedellec [9]
	Inductive Logic Programming	Syntax, Logic representation	Extract	Esposito et al. [8]
	Association rules	Syntax, Tokens	Extract	Maedche & Staab [24]
	Frequency-based	Syntax	Prune	Kietz et al. [19]
	Pattern-Matching		Extract	Morin [33]
	Classification	Syntax, Semantics	Refine	Schnattinger & Hahn [14, 4]
	Formal Concept Analysis	Syntax, Semantics	Refine	Basili ???? [14]
Dictionary	Information extraction	Syntax	Extract	Hearst [16], Wilks [51] and Kietz et al. [19]
	Page rank	Tokens		Jannink & Wiederhold [17]
Knowledge base	Concept Induction, A-Box mining	Relations	Extract	Kietz & Morik [20] and Schlobach [43]
Relational schemata	Data Correlation	Relations	Reverse engineering	Johannesson [18], Tari et al. [49], Stojanovic et al. [45]