Stephan Bloehdorn, Philipp Cimiano, Andreas Hotho, and Steffen Staab

An Ontology-based Framework for Text Mining

Structuring of text document knowledge frequently appears either by ontologies and metadata or by automatic (un-)unsupervised text categorization. This paper describes our integrated framework OTTO (OnTology-based Text mining framewOrk). OTTO uses text mining to learn the target ontology from text documents and uses then the same target ontology in order to improve the effectiveness of both supervised and unsupervised text categorization approaches.

1 Introduction

Most information resources available in the internet as well as within intranets are natural language text documents. It is often a prerequisite that these knowledge sources are structured in order to query for and retrieve them in a straightforward way. Speaking in very broad terms we recognize ongoing efforts for this purpose in two major directions.

First, researchers and practitioners working in the areas of information retrieval and text mining seek to find categories of textual resources by various fully automatic methods. The approaches either *(i)* predefine a metric on a document space in order to cluster 'nearby' documents into meaningful groups of documents (called 'unsupervised categorization' or 'text clustering'; Salton (1989)) or *(ii)* they adapt a metric on a document space to a manually predefined sample of documents assigned to a list of target categories such that new documents may be assigned to labels from the target list of categories, too ('supervised categorization' or 'text classification'; Sebastiani (2002)).

Second, researchers and practitioners working mainly in the areas of thesauri (Foskett 1997) and ontologies (Staab & Studer 2004) predefine conceptual structures and assign metadata to the documents that confirm to these conceptual structures.

Thereby, each of the two directions exhibits its advantages and problems. On the one hand the categorization of documents is (comparatively) cheap[1], but the

1 Automatic approaches are comparatively cheap even though the provisioning of sample data for supervised categorization may imply considerable, and sometimes even unbearable, costs.

quality of its document categorization for larger sets of target categories as well as the understandability of its results are often quite low. On the other hand, the quality of manual metadata may be very good, but the cost of building an ontology and adding manual metadata typically are one or several orders of magnitude higher than for automatic approaches.

To gain both advantages, while diminishing both their drawbacks at once, we here propose an approach of integrated ontology learning and text mining framework, viz. OTTO (OnTology-based Text mining framewOrk). Our implementation of OTTO includes a number of methods for (semi-)automatic ontology construction (also called *ontology learning*; Maedche & Staab (2004)) in order to provide for rich conceptual structures. Then, OTTO allows for exploitation of ontologies learned in this way by supervised or unsupervised text categorization.

We have shown in multiple contributions that ontology learning may be performed effectively (Maedche & Staab 2004; Cimiano et al. 2004b) and that text categorization may profit from ontologies (Bloehdorn & Hotho 2004; Hotho et al. 2003b, a). The integration we propose here allows for a tight integration of the two approaches combining their advantages.

The structure of the remainder of the paper is as follows: in Section 2 we introduce the overall OTTO text mining framework. In Section 3 we first present the TextToOnto system, which is designed to support the ontology engineer in the development of domain ontologies by applying text mining techniques. In this section we focus in particular on recent developments — as compared to Maedche & Staab (2004). In Section 4 we describe the approaches to text clustering and classification making use of ontologies as background knowledge. In Section 5 we discuss some related work and Section 6 concludes the paper.

2 General Architecture and Ontology Model

Figure 1 illustrates the overall OTTO system architecture. The architecture builds upon the Karlsruhe Ontology and Semantic Web Infrastructure (KAON)[2] that provides the access to implementations of our formal ontology model.

Ontology Model and Infrastructure KAON is a general and multi-functional open source ontology management infrastructure and tool suite developed

2 Forschungszentrum Informatik (FZI, WIM group, Karlsruhe) and Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB, Karlsruhe) (eds.) (2001-2005). KAON Homepage, `http://kaon.semanticweb.org` [accessed May 2005].

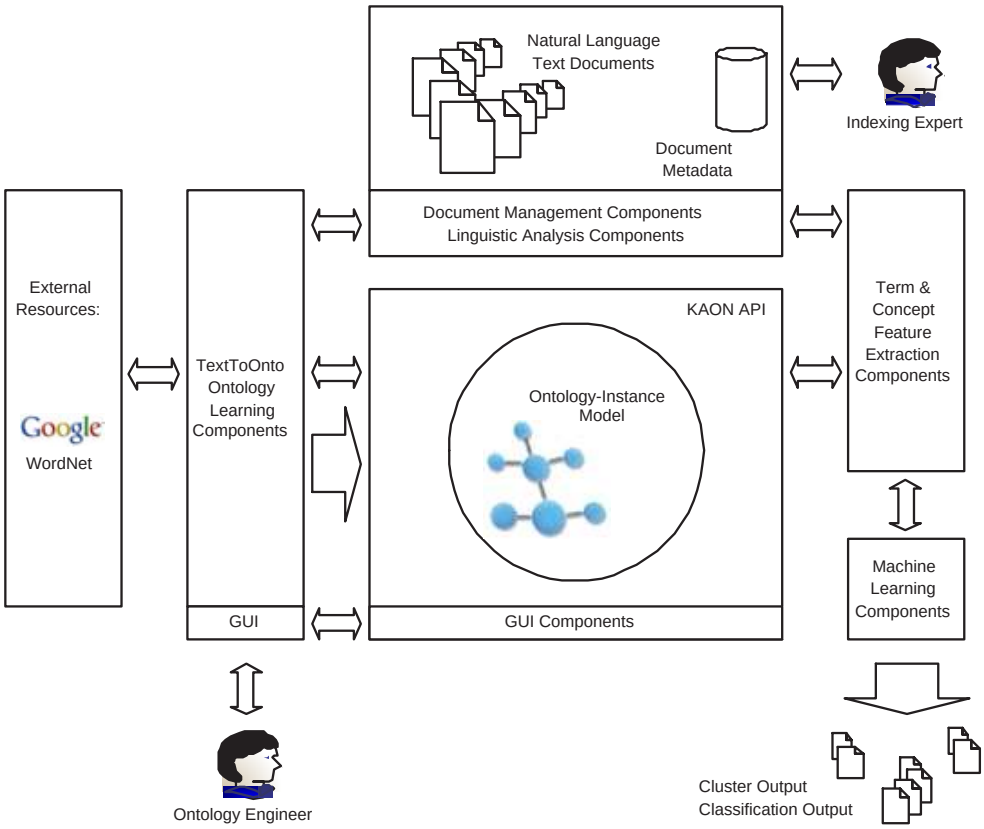

Figure 1: Overall OTTO System Architecture

at Karlsruhe University. KAON is built around the Ontology-Instance-Model (OI-model), a formal ontology model. In what follows we present our definition of an ontology which constitutes the formal model underlying an OI-model and we sketch the basic KAON system infrastructure. However, we only describe those parts of our more extensive ontology definition (E. Bozsak et al. 2002) that are needed for this paper.

Definition 2.1 (Core Ontology) *A* core ontology *is a structure*

$$\mathcal{O} := (C, \leq_C, R, \sigma, \leq_R)$$

consisting of two disjoint sets C and R whose elements are called concept identifiers *and* relation identifiers*, resp., a partial order* $\leq_C$ *on C, called* concept hierarchy *or* taxonomy*, a function* $\sigma\colon R \to C^+$ *called* signature*, a partial order* $\leq_R$ *on R, called* relation hierarchy*, where* $r_1 \leq_R r_2$ *implies* $|\sigma(r_1)| = |\sigma(r_2)|$ *and* $\pi_i(\sigma(r_1)) \leq_C \pi_i(\sigma(r_2))$*, for each* $1 \leq i \leq |\sigma(r_1)|$ *and* C^+ *is the set of tuples over C with at least one element and* π_i *is the i-th component of a given tuple.*

Definition 2.2 (Subconcepts and Superconcepts) *If $c_1 <_C c_2$ for any $c_1, c_2 \in C$, then c_1 is a* subconcept (specialization) *of c_2 and c_2 is a* superconcept (generalization) *of c_1. If $c_1 <_C c_2$ and there exists no $c_3 \in C$ with $c_1 <_C c_3 <_C c_2$, then c_1 is a direct subconcept of c_2, and c_2 is a direct superconcept of c_1, denoted by $c_1 \prec c_2$.*

The partial order $<_C$ relates the concepts in an ontology in form of specialization and generalization relationships, resulting in a hierarchical arrangement of concepts[3]. These relationships correspond to what is generally known as *is-a* or *is-a-special-kind-of* relations[4].

Often we will call concept identifiers and relation identifiers just *concepts* and *relations*, resp., for sake of simplicity. Almost all relations in practical use are binary. For those relations, we define their *domain* and their *range*.

Definition 2.3 (Domain and Range) *For a relation $r \in R$ with $|\sigma(r)| = 2$, we define its* domain *and its range by* $\text{dom}(r) := \pi_1(\sigma(r))$ *and* $\text{range}(r) := \pi_2(\sigma(r))$.

According to the international standard ISO 704, we provide names for the concepts (and relations). Instead of 'name', we here call them 'sign' or 'lexical entries' to better describe the functions for which they are used.

Definition 2.4 (Lexicon for an Ontology) *A* lexicon for an ontology *$\mathcal{O}$ is a tuple $Lex := (S_C, Ref_C)$ consisting of a set S_C, whose elements are called signs for concepts (symbols), and a relation $Ref_C \subseteq S_C \times C$ called lexical reference for concepts, where $(c, c) \in Ref_C$ holds for all $c \in C \cap S_C$. Based on Ref_C, for $s \in S_C$ we define $Ref_C(s) := \{c \in C | (s, c) \in Ref_C\}$. Analogously, for $c \in C$ it is $Ref_C^{-1}(c) := \{s \in S_C | (s, c) \in Ref_C\}$. An ontology with lexicon is a pair $(\mathcal{O}, Lex)$ where $\mathcal{O}$ is an ontology and Lex is a lexicon for $\mathcal{O}$.*

While the above definitions are related to the intensional and lexical aspects of an ontology, the following definition of a knowledge base relates to its extensional aspects:

Definition 2.5 (Knowledge Base) *A* knowledge base *is a structure*

$$KB := (C_{KB}, R_{KB}, I, \iota_C, \iota_R)$$

3 Note that this hierarchical structure is not necessarily a tree structure. It may also be a *directed acyclic graph* possibly linking concepts to multiple superconcepts at the same time.

4 In ontologies that are more loosely defined, the hierarchy may, however, not be as explicit as *is-a* relationships but rather correspond to the notion of *narrower-than* vs. *broader-than*. Note, however, that in many settings this view is considered as a very bad practice as it may lead to inconsistencies when reasoning with ontologies. However, this problem is not preeminent in the context of this work (Wielinga et al. 2001).

consisting of two sets C_{KB} and R_{KB}, a set I whose elements are called instance identifiers *(or* instances *or* objects *for short), a function $\iota_C : C_{KB} \rightarrow \mathfrak{P}(I)$ called* concept instantiation*, a function $\iota_R : R_{KB} \rightarrow \mathfrak{P}(I^+)$ with $\iota_R(r) \subseteq \prod_{c \in \sigma(r)} \iota_C(c)$, for all $r \in R$. The function ι_R is called* relation instantiation*,*

where $\mathfrak{P}(M)$ stands for the powerset of a set M and $\prod_i M_i$ for the cross-product of the sets M_i.

KAON features a full-fledged API that allows programmatic access to different implementations of the formal ontology model described. Currently, two different implementations of the KAON API are available: whereas the *KAON Engineering Server* is an ontology server using a scalable database representation of ontologies, *APIonRDF* is a main-memory implementation of the KAON API based on RDFS, a simple modelling language on top of the Resource Description Framework (RDF) formalism, both being developed by the W3C. *KAON OI-modeler* provides a graphical environment for ontology editing.

OTTO Text Mining Extensions OTTO's architecture is organized around KAON's OI-model and features various text mining modules (Figure 1). Separate document corpus management components allow to manage text document corpora and associated metadata information. Another core group of components offers basic linguistic analysis services like stemming, POS pattern analysis, word frequency calculations and the like, which are commonly used by all other components. The TextToOnto ontology learning algorithms, some of which will be described in section 3, can be applied to learn ontological structures from document corpora which are then stored in a corresponding OI-model. Some of the TextToOnto modules also make use of external resources like WordNet or `Google` in order to query the WWW. Comprehensible GUIs provide intuitive access to the learning algorithms as well as to the OI-model for the user. On the other hand, given that a suitable ontology is available, the OTTO concept extraction components allow to analyze text documents and extract a conceptual document representation that complements the classical bag-of-words document representation. We will have a closer look at these modules in section 3. The feature extraction components are carefully designed to allow flexible connections to different software modules that are capable to perform classical machine learning algorithms like classification or clustering. Implemented connectors include connectors to Weka[5], a Java based machine-learning library and application, or MATLAB.

5 Eibe, Frank et al. (eds.) (1999-2005). Weka 3: Data Mining Software in Java, `http://www.cs.waikato.ac.nz/~ml/weka/` [accessed May 2005].

3 The TextToOnto Ontology Learning Framework

TextToOnto[6] is a system conceived to support the ontology engineer in the task of creating and maintaining ontologies. For this purpose, it employs text mining techniques such as term clustering and matching of lexico-syntactic patterns as well as other resources of a general nature such as WordNet (Fellbaum 1998). In what follows, we describe the architecture as well as the algorithms used by the system to facilitate the ontology engineering process.

3.1 The TextToOnto Architecture

The main components of TextToOnto are the following (compare Maedche & Staab (2004) as well as Figure 2):

- The **Ontology Management Component** provides basic ontology management functionality. In particular, it supports editing, browsing and evolution of ontologies. For this purpose it builds upon the Karlsruhe Ontology and Semantic Web Infrastructure (KAON). In fact, KAON's OI-model is the key data structure on which the ontology learning process is centered.

- The **Algorithm Library Component** acts as the algorithmic backbone of the framework. It incorporates a number of text mining methods, e.g. conceptual clustering, terminology extraction, pattern matching as well as machine learning techniques, e.g. association rules and classifiers.

- **Coordination Component:** The ontology engineer uses this component to interact with the different ontology learning algorithms from the algorithm library. Comprehensive user interfaces are provided to select relevant corpora, set different parameters and start the various algorithms.

From a methodological point of view, the data structure around which the whole ontology learning process is centered is the OI-model as described in Section 2. The user can start with an empty OI-model and learn a new ontology from scratch or select an existing one and add new instances or relations. In this paper we do not describe all these components in detail, but refer the reader to Maedche & Staab (2004) instead. The main contribution of the present section is

6 The system is freely available and can be downloaded at Cimiano, Ph. et al. (eds.) (2003-2005). Project TextToOnto Homepage (Sourceforge), `http://sourceforge.net/projects/texttoonto/` [accessed May 2005].

in fact to present new components extending the functionalities of the system as described therein.

3.2 Ontology Learning Algorithms

In earlier work, we presented approaches for learning taxonomic relations via (i) top-down or bottom-up clustering techniques (Maedche et al. 2002; Cimiano et al. 2004b), (ii) matching lexico-semantic patterns or (iii) classification algorithms such as k-Nearest-Neighbours (Maedche & Staab 2002). Further, we also developed algorithms for extracting general binary relations between concepts based on association rules mining (Maedche & Staab 2000). Another possibility we examined is to extract domain ontologies from large, domain independent ontologies by pruning (Volz et al. 2003). In this paper we present three new algorithms actually implemented within TextToOnto with the purpose of:

- constructing taxonomies using a conceptual clustering algorithm, i.e. Formal Concept Analysis (TaxoBuilder component)
- constructing taxonomies by combining information aggregated from WordNet, Hearst (Hearst 1992) patterns matched in a corpus as well as certain heuristics (TaxoBuilder component)
- classifying instances into the ontology by using lexico-syntactic patterns (InstanceExtraction component)
- extracting labelled relations and specifying their domain and range (RelationLearning component)

3.2.1 TaxoBuilder

TaxoBuilder is a component developed for the purpose of learning concept hierarchies from scratch. It can be used in two different modes:

- In **FCA** mode, TaxoBuilder employs the technique described in Cimiano et al. (2004a) to learn a concept hierarchy by means of Formal Concept Analysis (Ganter & Wille 1999).
- In **Combination** mode, TaxoBuilder uses different sources of evidence such as WordNet, Hearst patterns (Hearst 1992) matched in a corpus as well as certain heuristics to find taxonomic relations.

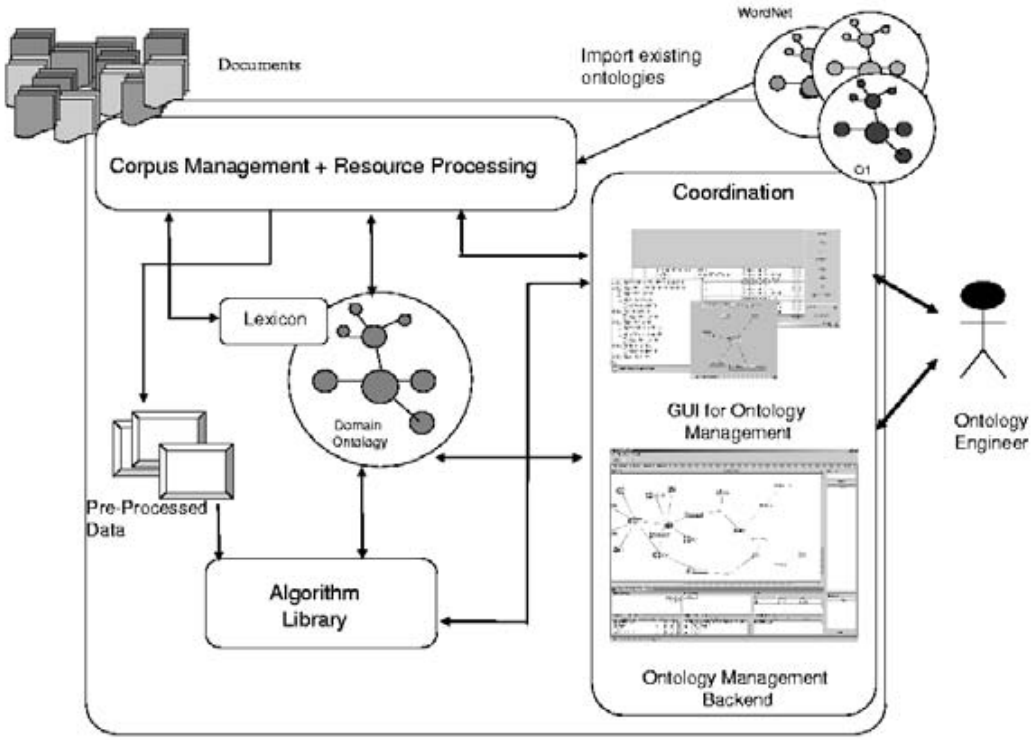

Figure 2: TextToOnto Architecture

In the FCA mode, TaxoBuilder extracts syntactic dependencies from text by applying shallow parsing techniques. In particular it extracts verb-object relations and uses them as context attributes for Formal Concept Analysis as described in Cimiano et al. (2004a) and Cimiano et al. (2004b). The lattice is then built in the background and transformed into an OI-model by removing the bottom *formal concept* and introducing for every formal concept an ontological concept named with its intent. For every element in the extension of this formal concept we introduce an ontological subconcept. Figure 3 shows for example the lattice automatically learned for the following terms: *apartment, hotel, car, bike* and *trip*. The corresponding formal context is depicted in Table 1. As already mentioned, the lattice is calculated in the background and transformed into the OI-model in Figure 4.

This approach has been evaluated in Cimiano et al. (2004a) and Cimiano et al. (2004b) by comparing the automatically generated concept hierarchies with handcrafted hierarchies for a given domain in terms of the similarity measures described in Maedche & Staab (2002).

	runable	offerable	needable	startable	meanable	seemable	attemptable	cruiseable	fillable
apartment	x								
hotel	x	x							
car			x						
bike				x					
trip					x	x	x	x	x

Table 1: Formal Context for the terms *apartment, hotel, car, bike* and *trip*

In the *Combination* mode, TaxoBuilder exploits (i) the vertical relations heuristic in Missikoff et al. (2002), (ii) Hearst patterns (Hearst 1992) as well as (iii) the hypernym relations in WordNet (Fellbaum 1998). Now given a pair of terms, say t_1 and t_2, they could be taxonomically related in two ways: is-a(t_1,t_2) or is-a(t_2,t_1). In order to decide in which way they are related we compute the evidence for both of the relations by taking into account the above information sources. In particular, we take into account the above mentioned heuristic, the number of Hearst patterns in the corpus found as well as the number of hypernymic paths in WordNet between two terms. We sum up all these values and choose the relation with maximum evidence. All the taxonomic relations found in this way between a given set of terms in question are then added to the OI-model after removing potential cycles. This method has been proven to be an effective way of quickly learning concept hierarchies. Figure 5 shows a concept hierarchy automatically acquired with the combination method out of 500 texts from the online *Lonely Planet* world guide[7].

3.2.2 InstanceExtraction

The InstanceExtraction component discovers instances of concepts of a given ontology in a text corpus. So, it needs a text corpus and a non-empty OI-model as input. It can either be used in a semi-automatic or fully automatic way. In the first case, it will present the candidate instances to the user asking for confirmation, while in the second case it will simply add the discovered instances to the corresponding OI-model. In order to discover these instances, InstanceExtraction makes use of a combination of patterns from Hearst (1992) and Hahn & Schnattinger (1998). The user can choose which of the different patterns s/he wants to use. The patterns are described in what follows:

7 Lonely Planet Publications (2005). Lonely Planet Homepage, `http://www.lonelyplanet.com` [accessed May 2005].

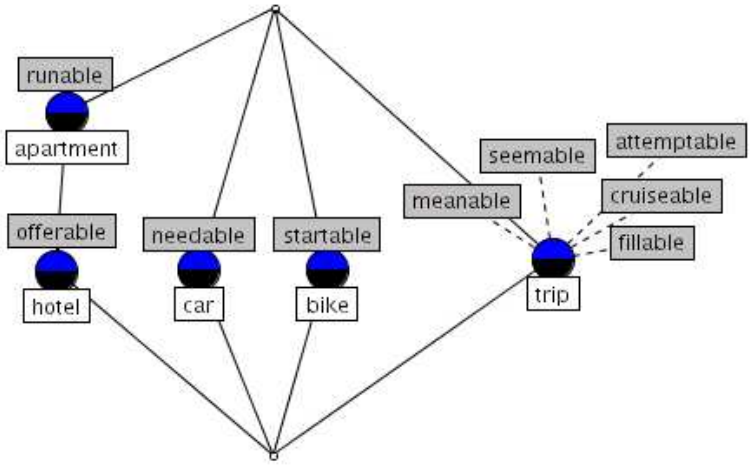

Figure 3: Concept Lattice

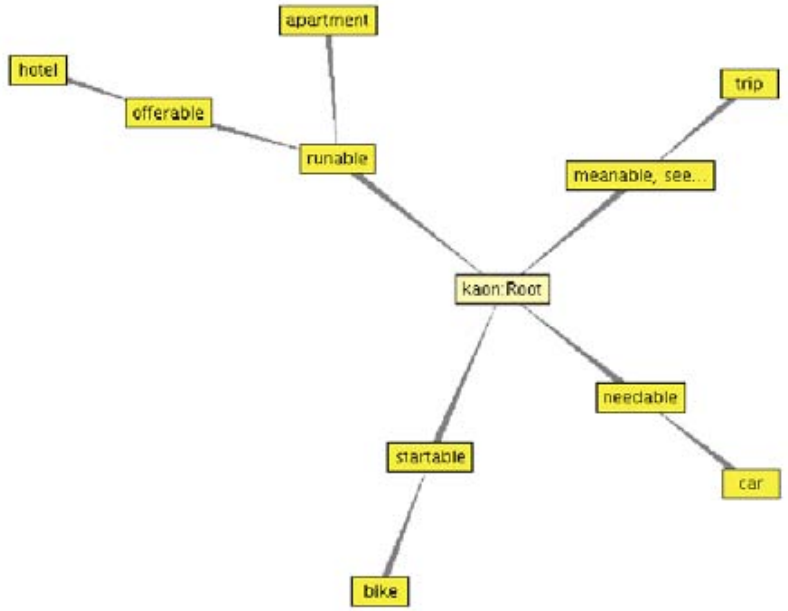

Figure 4: OI-model automatically learned with the FCA approach

Hearst Patterns The first four patterns have been used by Hearst to identify *is-a*-relationships between the concepts referred by two terms in the text. However, they can also be used to categorize a named entity or instance into an ontology. In our approach we have the underlying assumption that common nouns represent concepts and proper nouns represent instances. In order to

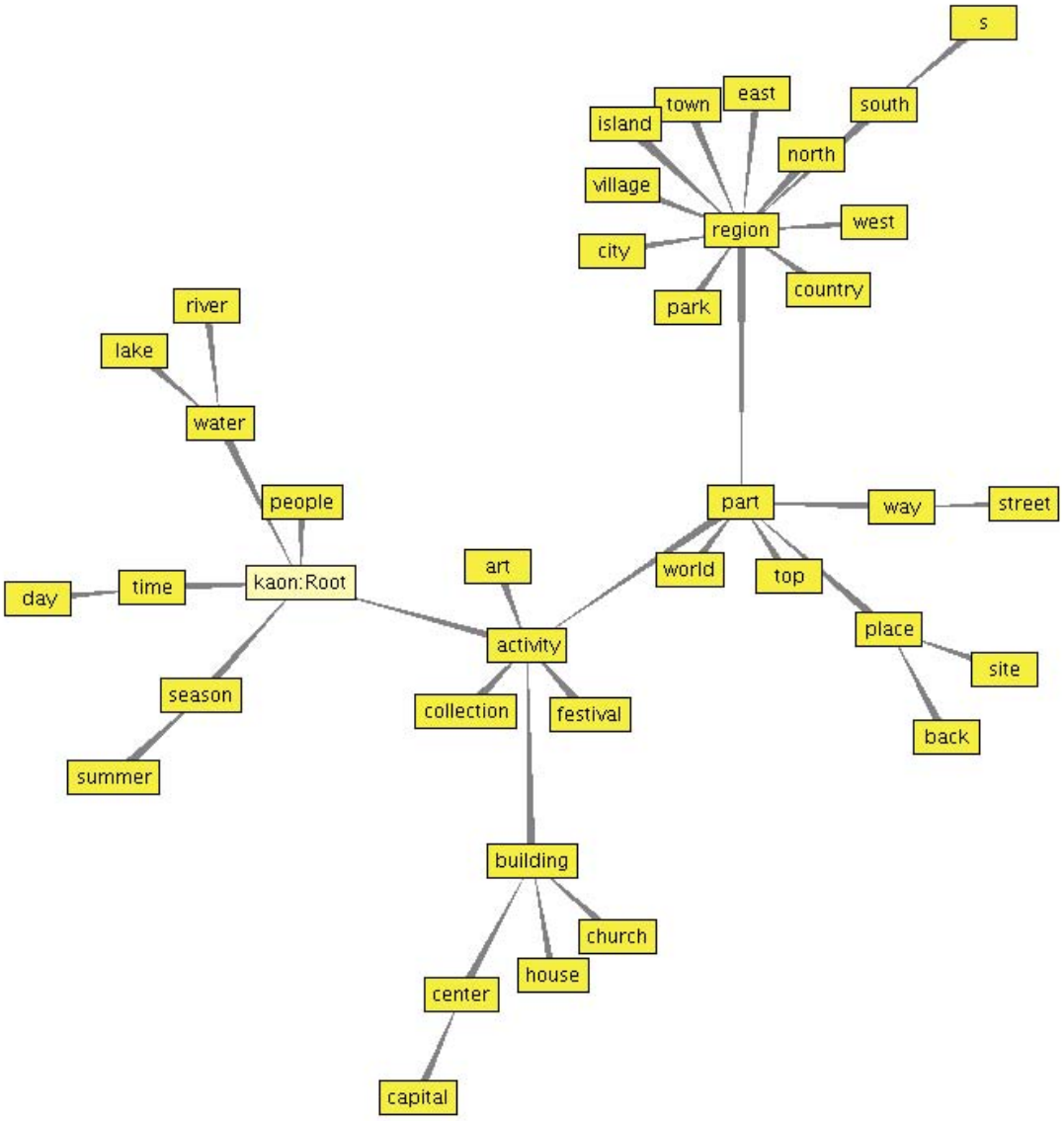

Figure 5: OI-model automatically learned with the combination approach

identify noun phrases representing concepts, henceforth $NP_{CONCEPT}$, and noun phrases representing instances, henceforth $NP_{INSTANCE}$, we use a shallow parsing technique based on matching regular expressions over part-of-speech tags to identify the two types of noun phrases described above. The patterns reused from Hearst are:

HEARST1: $NP_{CONCEPT}$ such as $NP_{INSTANCE}$

HEARST2: such $NP_{CONCEPT}$ as $NP_{INSTANCE}$

HEARST3: $NP_{CONCEPT}$, (especially|including) $NP_{INSTANCE}$

HEARST4: $NP_{INSTANCE}$ (and|or) other $NP_{CONCEPT}$

The above patterns would match the following expressions (in this order): *hotels such as Ritz*; *such hotels as Hilton*; *presidents, especially George Washington*; and *the Eiffel Tower and other sights in Paris*.

Definites The next patterns are about definites, i.e. noun phrases introduced by the definite determiner '*the*'. Frequently, definites actually *refer* to some entity previously mentioned in the text. In this sense, a phrase like '*the hotel*' does not stand for itself, but it points as a so-called anaphora to a unique hotel occurring in the preceding text. Nevertheless, it has also been shown that in common texts more than 50% of all definite expressions are *non-referring*, i.e. they exhibit sufficient descriptive content to enable the reader to uniquely determine the entity referred to from the global context (Poesio & Vieira 1998). For example, the definite description '*the Hilton hotel*' has sufficient descriptive power to uniquely pick-out the corresponding real-world entity for most readers. One may deduce that '*Hilton*' is the name of the real-world entity of type hotel to which the above expression refers.

Consequently, we apply the following two patterns to categorize candidate proper nouns by definite expressions:

DEFINITE1: the $NP_{INSTANCE}$ $NP_{CONCEPT}$

DEFINITE2: the $NP_{CONCEPT}$ $NP_{INSTANCE}$

The first and the second pattern would, e.g., match the expressions '*the Hilton hotel*' and '*the hotel Hilton*', respectively.

Apposition and Copula The following pattern makes use of the fact that certain entities appearing in a text are further described in terms of an apposition as in '*Excelsior, a hotel in the center of Nancy*'. The pattern capturing this intuition looks as follows:

APPOSITION: $NP_{INSTANCE}$, a $NP_{CONCEPT}$

The probably most explicit way of expressing that a certain entity is an instance of a certain concept is by the verb '*to be*', as for example in '*The Excelsior is a hotel in the center of Nancy*'. Here's the general pattern:

COPULA: $NP_{INSTANCE}$ is a $NP_{CONCEPT}$

Pattern	Suggested	Annotator 1	Annotator 2	Annotator 3	Accuracy
HEARST1	2	40.00%	40.00%	60.00%	46.66%
DEFINITE1	19	21.05%	36.84%	36.84%	31.56%
DEFINITE2	74	91.36%	93.83%	96.30%	93.83%
APPOSITION	28	56.00%	62.00%	62.00%	60.00%
COPULA	22	66.67%	66.67%	63.64%	65.66%
ALL	188	69.15%	73.40%	74.47%	72.34%

Table 2: Accuracy of each of the patterns

Evaluation In order to evaluate our pattern-based approach to categorizing instances, we considered the 500 randomly selected web pages from *Lonely Planet* and used a part-of-speech (POS) tagger[8] as well handcrafted rules to match non-recursive NPs representing concepts and instances, respectively, as well as the above patterns.
We then presented the found instance-concept pairs to three different subjects for validation. They had the possibility of validating the relationship, adding the concept name to the instance, rejecting the relationship or expressing their doubt. The possibility of adding the concept name is important when judging a suggestion such as that *Lenin* is an instance of a *museum*. In this case, the users could decide that the suggestion of the system is not totally wrong and correct the suggestion by specifying that *Lenin museum* is the actual instance of a *museum*. In this case we counted the answer of the system as correct. Table 2 gives the accuracy for all the patterns based on the answers of the human subjects to the suggestions of the system. Unfortunately, no HEARST2, HEARST3 or HEARST4 instances were found in the texts, which shows that they are actually the ones which occur most rarely. Interestingly, it can be appreciated that the accuracy varies from pattern to pattern. Overall, the performance of the approach seems very reasonable as more than 72% of the suggested relations are judged as correct by the human subjects.

3.2.3 RelationLearning

The RelationLearning component also discovers candidate relations from text but in contrast to the association rule algorithm described in Maedche & Staab (2004) suggests a name for the relation to the user as well as *domain* and *range* for it. For this purpose, it employs a shallow parsing strategy to extract subcate-

8 We used the QTag POS-Tagger, cf. Mason, O. (1994-2003). QTag Homepage, `http://www.english.bham.ac.uk/staff/omason/software/qtag.html` [accessed May 2005].

gorization frames enriched with selectional restrictions specified with regard to the corresponding OI-model as described in Resnik (1997). In particular, it extracts the following syntactic frames:

- transitive, e.g. love(subj,obj)
- intransitive + PP-complement, e.g. walk(subj,pp(to))
- transitive + PP-complement, e.g. hit(subj,obj,pp(with))

RelationLearning then enriches these subcategorization frames semantically by finding the appropriate concept from a given ontology for each syntactic position. For each occurrence of a given syntactic frame, it extracts the nominal head in each syntactic position and augments the corresponding concept count by one. For each syntactic frame and syntactic position it chooses the most specific concept with maximal count. On the basis of these subcategorization frames, it suggests possible relations to the user for validation. For example, given the following enriched subcategorization frames

love(subj:*person*,obj:*person*)
walk(subj:*person*,to:*place*)
hit(subj:*person*,obj:*thing*,with:*contundent_object*)

the system would suggest the following relations to the user:

love(domain:*person*,range:*person*)
walk_to(domain:*person*,range:*place*)
hit(domain:*person*,range:*thing*)
hit_with(domain:*person*,range:*contundent_object*)

The main problem with this approach to discovering relations is related to data sparseness as for small to medium-sized corpora there are not enough verbs in the text collection connecting all the different concepts of the ontology together. In general with this approach we thus end up with only a small number of relations.

4 Ontology-based Text Clustering and Classification

Due to the ever growing amounts of textual information available electronically, users are facing the challenge of organizing, analyzing and searching large

numbers of documents. Systems that automatically classify text documents into predefined thematic classes or detect clusters of documents with similar content offer a promising approach to tackle this complexity. During the last decades, a large number of machine learning algorithms have been proposed for supervised and unsupervised text categorization. So far, however, existing text categorization systems have typically used the *Bag–of–Words model* known from information retrieval, where single words or word stems are uses as features for representing document content (Salton 1989). In this section we present an approach that exploits existing ontologies by using their lexica and concept hierarchies to improve results in both, supervised and unsupervised settings.

4.1 The Bag-of-Words Model

In the *Bag–of–Words paradigm*, documents are represented as bags of terms. Let D be the set of documents and $T = \{t_1, \ldots, t_m\}$ the set of all different terms occurring in D. The absolute frequency of term $t \in T$ in document $d \in D$ is given by $\mathrm{tf}(d, t)$. Term vectors are denoted $\vec{t_d} = (\mathrm{tf}(d, t_1), \ldots, \mathrm{tf}(d, t_m))$.

Stopwords and Stemming The initial term vectors produced so far in this approach can be further modified as described in what follows. *Stopwords* are words which are considered as non–descriptive within a bag–of–words approach. For example, for english language, it is common practice to use a standard list of 571 stopwords initially designed for the SMART system[9]. Typically, text documents are further processed to reduce the term representation to term stems, e.g. using the Porter stemmer introduced in Porter (1980). Using *stemmed terms*, one can construct a vector representation $\vec{t_d}$ for each text document.

Pruning Pruning rare terms also affects results. Depending on a pre-defined threshold δ, a term t is discarded from the representation (i. e., from the set T), if $\sum_{d \in D} \mathrm{tf}(d, t) \leq \delta$. In our experiments, we have for example used the values 0, 5 and 30 for δ. The rationale behind *pruning* is that infrequent terms do not help for identifying appropriate clusters, but may still add noise to the distance measures degrading overall performance.

Weighting Having extracted the collection of terms that make up the documents in a corpus, the corresponding numeric values of the terms within

9 SMART Project (eds.) Stopword List for English Information Retrieval, `http://www.unine.ch/info/clef/englishST.txt` [accessed May 2005].

the document have to be determined. A special case of term weighting is binary weighting, where the terms are represented as boolean variables. *Tfidf* weighs the frequency of a term in a document with a factor that discounts its importance when it appears in almost all documents. The tfidf (term frequency−inverted document frequency)[10] of term t in document d is defined by: $\text{tfidf}(d,t) := \log(\text{tf}(d,t)+1) * \log\left(\frac{|D|}{\text{df}(t)}\right)$ where $\text{df}(t)$ is the document frequency of term t that counts in how many documents term t appears. If tfidf weighting is applied then we replace the term vectors $\vec{t_d} := (\text{tf}(d,t_1),\ldots,\text{tf}(d,t_m))$ by $\vec{t_d} := (\text{tfidf}(d,t_1),\ldots,\text{tfidf}(d,t_m))$. There are more sophisticated measures than tfidf in the literature (see, e. g., Amati et al. (2001)), but we abstract herefrom, as this is not the main topic of this paper.

Deficiencies By using only single terms to represent document content any chosen machine learning algorithm is restricted to detecting patterns in the used *terminology* only, while *conceptual* patterns remain ignored. Specifically, systems using only words as features exhibit a number of inherent deficiencies:

1. *Multi-Word Expressions* with an own meaning like *"European Union"* are chunked into pieces with possibly very different meanings like *"union"*.

2. *Synonymous Words* like *"tungsten"* and *"wolfram"* are mapped into different features.

3. *Polysemous Words* are treated as one single feature while they may actually have multiple distinct meanings.

4. *Lack of Generalization*: there is no way to generalize similar terms like "beef" and "pork" to their common hypernym "meat".

While items 1 – 3 directly address issues that arise on the lexical level, items 4 rather addresses an issue that occurs at the conceptual level.

In our approach, we use background knowledge in form of simple ontologies (cf. section 2) to improve text classification and clustering results by directly addressing these problems. We propose a hybrid approach for document representation based on the common term stem representation which is enhanced with concepts extracted from the used ontologies.

10 tfidf actually refers to a class of weighting schemata. Above we have given the one we have used.

4.2 Enriching the Document Vectors with Cconcepts

In our approach, we exploit background knowledge about concepts that is explicitly given according to our ontological model (cf. section 2). For this purpose, we extend each term vector $\vec{t_d}$ by new entries for ontological concepts c appearing in the document set. Thus, the vector $\vec{t_d}$ is replaced by the concatenation of $\vec{t_d}$ with the concept vector $\vec{c_d} := (\mathrm{cf}(d, c_1), \ldots, \mathrm{cf}(d, c_l))$ having length $l = |C|$ and where $\mathrm{cf}(d, c)$ denotes the frequency of the appearance of concept $c \in C$ in document d as indicated by applying the reference function Ref_C to all terms in the document d. Hence, a term that also appears in the ontology would be accounted for at least twice in the new vector representation, i. e., once as part of the old $\vec{t_d}$ and at least once as part of $\vec{c_d}$. It could be accounted for also more often, because a term like "bank" has several corresponding concepts in the ontology.

To extract the concepts from texts, we have developed a detailed process, that can be used with any ontology with lexicon. The overall process comprises five processing steps that are described in the following.

1. Candidate Term Detection Due to the existence of multi-word expressions, the mapping of terms to concepts can not be accomplished by querying the lexicon directly for the single words in the document.

We have addressed this issue by developing a candidate term detection algorithm (Bloehdorn & Hotho 2004) that builds on the basic assumption that finding the longest multi-word expressions that appear in the text and the lexicon will lead to a mapping to the most specific concepts. The algorithm works by moving a window over the input text, analyzing the window content and either decreasing the window size if unsuccessful or moving the window further. For English, a window size of 4 is sufficient to detect virtually all multi-word expressions.

2. Syntactical Patterns Querying the lexicon directly for any expression in the window will result in many unnecessary searches and thereby in high computational requirements. Luckily, unnecessary search queries can be identified and avoided through an analysis of the part-of-speech (POS) tags of the words contained in the current window. Concepts are typically symbolized in texts within *noun phrases*. By defining appropriate POS patterns and matching the window content against these, multi-word combinations that will surely not

symbolize concepts can be excluded in the first hand and different syntactic categories can be disambiguated.

3. Morphological Transformations Typically the lexicon will not contain all inflected forms of its entries. If the lexicon interface or separate software modules are capable of performing base form reduction on the submitted query string, queries can be processed directly. For example, this is the case with WordNet. If the lexicon, as in most cases, does not contain such functionalities, a simple fallback strategy can be applied. Here, a separate index of stemmed forms is maintained. If a first query for the inflected forms on the original lexicon turned out unsuccessful, a second query for the stemmed expression is performed.

4. Word Sense Disambiguation Having detected a lexical entry for an expression, this does not necessarily imply a one-to-one mapping to a concept in the ontology. Although multi-word-expression support and POS pattern matching reduce ambiguity, there may arise the need to disambiguate an expression versus multiple possible concepts. The *word sense disambiguation (WSD)* task is a problem in its own right (Ide & Véronis 1998) and was not the focus of our work.

In our experiments, we have used three simple strategies proposed in Hotho et al. (2003c) to process polysemous terms:

- The ''all'' strategy leaves actual disambiguation aside and uses all possible concepts.
- The ''first'' strategy exploits WordNet's capability to return synsets ordered with respect to usage frequency. This strategy chooses the most frequent concept in case of ambiguities.
- The ''context'' strategy performs disambiguation based on the degree of overlap of lexical entries for the semantic vicinity of candidate concepts and the document content as proposed in Hotho et al. (2003c).

5. Generalization The last step in the process is about going from the specific concepts found in the text to more general concept representations. However, we do not only add the concepts directly representing the terms but also the corresponding superconcept along the path to the root of the concept hierarchy. An important issue here is to restrict the number of levels up in the hierarchy considered for adding superconcepts. The following procedure realizes this idea

by adding to the concept frequency of higher level concepts in a document d the frequencies of their subconcepts (of at most r levels down in the hierarchy). I. e., the vectors we consider are first of the form $\vec{t_d} := (\mathrm{tf}(d,t_1),\ldots,\mathrm{tf}(d,t_m),$ $\mathrm{cf}(d,c_1),\ldots,\mathrm{cf}(d,c_n))$ (the concatenation of an initial term representation with a concept vector). Then the frequencies of the concept vector part are updated, for a user-defined $r \in \mathbb{N}_0$, in the following way: For all $c \in C$, replace $\mathrm{cf}(d,c)$ by $\mathrm{cf}'(d,c) := \sum_{b \in H(c,r)} \mathrm{cf}(d,b)$, where $H(c,r) := \{c' | \exists c_1,\ldots,c_i \in C\colon c' \prec c_1 \prec \ldots \prec c_i = c,\ 0 \leq i \leq r\}$ gives for a given concept c the r next subconcepts in the taxonomy. In particular $H(c,\infty)$ returns all subconcepts of c. This implies: The strategy $r = 0$ does not change the given concept frequencies, $r = n$ adds to each concept the frequency counts of all subconcepts in the n levels below it in the ontology and $r = \infty$ adds to each concept the frequency counts of all its subconcepts.

4.3 Machine Learning Components and Results

As documents have been processed with the term and concept extraction components, they can be processed using standard machine learning algorithms. Currently, we use an interface that allows easy integration of the resulting hybrid document feature representations into WEKA[11], a Java-based multi-purpose machine learning environment.

Unsupervised Text Categorization (Clustering) deals with grouping documents together that are homogenous in some way. In contrast to supervised text categorization, where the classes in question are assigned outside the learning environment, it is the very task of the clustering algorithm to find good groups (clusters) in the first hand when no classes are given a priori.

For clustering (Steinbach et al. 2000), it has been shown that Bi-Section-KMeans – a variant of KMeans – frequently outperforms standard KMeans as well as agglomerative clustering techniques. Thus, we make use of Bi-Section-KMeans as clustering method. The similarity between two text documents $d_1, d_2 \in D$ is measured by the cosine of the angle between the vectors $\vec{t}_1, \vec{t}_2$ representing them:

$$\cos(\sphericalangle(\vec{t}_1,\vec{t}_2)) = \frac{\vec{t}_1 \cdot \vec{t}_2}{\| \vec{t}_1 \| \cdot \| \vec{t}_2 \|}$$

In experiments reported in a previous paper (Hotho et al. 2003c), we showed that conceptual representations can significantly improve text cluster purity by

11 See footnote 5 above.

reducing the variance among the representations within the given classes of related documents. In the experiments on the well-known Reuters-21578 corpus using WordNet as ontology, we were able to show a significant improvement of up to 8% using a simple word sense disambiguation strategy combined with generalization based on term and concept vectors. We observed a performance drop without using any word sense disambiguation. An investigation of the different clusters revealed that some given classes of the Reuters corpus could be found with a high purity by the clustering algorithm while for other classes purity decreases.

Supervised Text Categorization Not surprisingly, supervised text categorization and clustering are closely related as both are concerned with "groupings" of objects. However, in the supervised setting, these groupings are given by the common membership to a thematic class that is assigned to sample documents before the training process starts. The training process then induces hypotheses of how the document space is shaped according to which new documents are assigned target categorizations.

Many different supervised categorization algorithms have been designed and virtually all of them have been used for text categorization tasks, including probabilistic classifiers like Naïve Bayes, Linear Discriminant Functions like Perceptrons or more recently Support Vector Machines, Decision Trees and Decision Rule Classifiers, Nonparametric Classifiers like k-Nearest-Neighbours and Ensemble Classifiers, most namely Bagging and Boosting. Comparisons like in Sebastiani (2002) suggest that *Boosting* (Schapire & Singer 2000) and *Support Vector Machines* (Joachims 1998) are the most promising approaches for handling text classification tasks.

In a recent experimental evaluation on two well-known text corpora (Bloehdorn & Hotho 2004), the Reuters-21578 corpus and the medical document corpus OHSUMED, were able to show the positive effects of our approach. Using Boosting as actual learning algorithm and both, term stems and concepts as features, we were able to achieve consistent improvements of the categorization results. In terms of the well-known F_1 measure, that combines precision and recall results this improvement was in the 1% – 3% range for the Reuters-21578 corpus and in the 2.5% – 7% range for the OHSUMED corpus[12]. The difference between both evaluations is probably explained best by the fact that the medical documents in the OHSUMED corpus make heavy use of multi-word-

12 These figures are based on *macro-averaged* F_1 results with *micro-averaged* results being slightly worse on the Reuters-21578 corpus while being fairly similar on the OHSUMED corpus.

expressions, synonyms and very specific terms which obfuscates a pure term based representation very much, while conceptual features tend to reduce noise in these situations.

5 Related Work

In this section we discuss work related to text mining techniques for ontology learning as well as text clustering and classification techniques relying on background knowledge.

Ontology Learning There is quite a long tradition in learning concept hierarchies by clustering approaches such as the ones presented in Hindle (1990); Pereira et al. (1993); Faure & Nedellec (1998); Caraballo (1999); Bisson et al. (2000) as well as by matching lexico-syntactic patterns as described in Hearst (1992, 1998); Charniak & Berland (1999); Poesio et al. (2002); Ahmid et al. (2003); Jouis (1993); Seguela (2001); Cimiano et al. (2004). In this section we focus on the discussion of frameworks and systems designed for supporting the ontology engineering process. In the ASIUM system (Faure & Nedellec 1998) nouns appearing in similar contexts are iteratively clustered in a bottom-up fashion. In particular, at each iteration, the system clusters the two most similar extents of some argument position of two verbs and asks the user for validation. Bisson et al. (2000) present an interesting framework and a corresponding workbench - Mo'K - allowing users to design conceptual clustering methods to assist them in an ontology building task. The framework is general enough to integrate different clustering methods. Velardi et al. (2001) present the OntoLearn system which discovers i) the domain concepts relevant for a certain domain, i.e. the relevant terminology, ii) named entities, iii) 'vertical' (is-a or taxonomic) relations as well as iv) certain relations between concepts based on specific syntactic relations. In their approach a 'vertical' relation is established between a term t_1 and a term t_2, i.e. is-a(t_1,t_2), if the head of t_2 matches the head of t_1 and additionally the former is additionally modified in t_1. Thus, a 'vertical' relation is for example established between the term 'international credit card' and the term 'credit card', i.e. is-a(international credit card,credit card).

Background Knowledge for Text Categorization Tasks To date, the work on integrating semantic background knowledge into text categorization is quite scattered. Much of the early work with semantic background knowledge in infor-

mation retrieval was done in the context of *query expansion* techniques (Bodner & Song 1996). Others like Green (1999) or Kushal Dave (2003) were more or less successful in using WordNet synsets to improve the text clustering task. Further they only investigate the use of WordNet and not ontologies in general by only applying a small number of strategies of the kind that we have investigated.

Recent experiments with conceptual feature representations for supervised text categorization are presented in Wang et al. (2003). These and other similar published results are, however, still too few to allow insights on whether positive results can be achieved in general. In some cases, even negative results were reported. For example, a comprehensive comparison of approaches with different document representations based on word senses and different learning algorithms ends with the conclusion of the authors that *"the use of word senses does not result in any significant categorization improvement"* (Kehagias et al. 2000). While we have been able to confirm the results they achieved for their method inventory, we have also shown that an enriched set of methods improves results by a large margin. In particular, we have found that ontology-based approaches benefit from feature weighting and word sense disambiguation.

Alternative approaches for conceptual representations of text documents that are not based on background knowledge compute kind of "statistical" concepts. Very good results with a probabilistic variant of LSA known as Probabilistic Latent Semantic Analysis (pLSA) were recently reported in Cai & Hofmann (2003). The experiments reported therein are of particular interest as the classification was also based on AdaBoost and was also using a combined term-concept representation, the latter being however automatically extracted from the document corpus using pLSA. We have investigated some of these approaches. We have been able to show that indeed LSA improves text clustering. In addition, we could show that ontology based approaches further improve the results achieved by LSA. Further comparisons with pLSA remain to be done in future research.

6 Conclusion and Further Work

Exploiting knowledge present in textual documents is an important issue in building systems for knowledge management and related tasks. In this paper we have presented OTTO (OnTology-based Text mining framewOrk), a framework centered around the KAON OI-model for the interaction between ontologies, i.e. explicit formalizations of a shared conceptualization and natural language texts in two directions.

First, natural language processing techniques combined with machine learning algorithms allow to build or extend ontologies in a semi-automatic manner. This field, known as ontology learning, is critical for building domain specific ontologies with fewer manual effort. We have presented recent innovations in this field that have been implemented in the TextToOnto modules of our OTTO framework.

Second, background knowledge in form of ontologies enhances the performance of classical text mining tasks such as text classification and text clustering. Semantic features extracted from ontologies with help of the OTTO text mining components leverage the classical bag–of–words representation to a higher semantic level and thereby improve classification accuracy and cluster purity.

Future work in this area will focus on a more thorough analysis how domain ontologies learned by means of ontology learning techniques can improve text classification and clustering tasks on documents from the same corpus, compared to using general purpose ontologies or linguistic resources like WordNet. Preliminary results show that this is a promising approach and will heavily influence the design of future OTTO module extensions.

References

Ahmid, K., Tariq, M., Vrusias, B., & Handy, C. (2003). Corpus-based thesaurus construction for image retrieval in specialist domains. In *Proceedings of the 25th European Conference on Advances in Information Retrieval (ECIR)*.

Amati, G., Carpineto, C., & Romano, G. (2001). Fub at trec-10 web track: A probabilistic framework for topic relevance term weighting. In *TREC 2001*. online publication.

Bisson, G., Nedellec, C., & Canamero, L. (2000). Designing clustering methods for ontology building - The Mo'K workbench. In *Proceedings of the ECAI Ontology Learning Workshop*.

Bloehdorn, S. & Hotho, A. (2004). Boosting for Text Classification with Semantic Features. In *Proceedings of the MSW 2004 workshop at the 10th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* .

Bodner, R. C. & Song, F. (1996). Knowledge-Based Approaches to Query Expansion in Information Retrieval. In *Advances in Artificial Intelligence*. New York, NY, USA: Springer.

Cai, L. & Hofmann, T. (2003). Text Categorization by Boosting Automatically Extracted Concepts. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, Toronto, Canada. ACM Press.

Caraballo, S. (1999). Automatic construction of a hypernym-labeled noun hierarchy from text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, (pp. 120–126).

Charniak, E. & Berland, M. (1999). Finding parts in very large corpora. In *Proceedings of the 37th Annual Meeting of the ACL*, (pp. 57–64).

Cimiano, P., Hotho, A., & Staab, S. (2004a). Clustering ontologies from text. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*.

Cimiano, P., Hotho, A., & Staab, S. (2004b). Comparing conceptual, divisive and agglomerative clustering for learning taxonomies from text. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*.

Cimiano, P., Pivk, A., Schmidt-Thieme, L., & Staab, S. (2004). Learning taxonomic relations from heterogeneous evidence. In *Proceedings of the ECAI'04 Workshop on Ontology Learning and Population*.

E. Bozsak et al. (2002). KAON - Towards a large scale Semantic Web. In *Proceedings of the Third International Conference on E-Commerce and Web Technologies (EC-Web)*. Springer Lecture Notes in Computer Science.

Faure, D. & Nedellec, C. (1998). A corpus-based conceptual clustering method for verb frames and ontology. In Velardi, P. (Ed.), *Proceedings of the LREC Workshop on Adapting lexical and corpus resources to sublanguages and applications*.

Fellbaum, C. (1998). *WordNet, an electronic lexical database*. MIT Press.

Foskett, D. J. (1997). Thesaurus. In P. Willett & K. Sparck-Jones (Eds.), *Reproduced in Readings in Information Retrieval* (pp. 111–134). Morgan Kaufmann.

Ganter, B. & Wille, R. (1999). *Formal Concept Analysis – Mathematical Foundations*. Springer Verlag.

Green, S. J. (1999). Building hypertext links by computing semantic similarity. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, *11*(5), 713–730.

Hahn, U. & Schnattinger, K. (1998). Towards text knowledge engineering. In *AAAI'98/IAAI'98 Proceedings of the 15th National Conference on Artificial Intelligence and the 10th Conference on Innovative Applications of Artificial Intelligence*.

Hearst, M. (1992). Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*.

Hearst, M. (1998). Automated discovery of wordnet relations. In C. Fellbaum (Ed.), *WordNet: An Electronic Lexical Database*. MIT Press.

Hindle, D. (1990). Noun classification from predicate-argument structures. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, (pp. 268–275).

Hotho, A., Staab, S., & Stumme, G. (2003a). Explaining text clustering results using semantic structures. In *Principles of Data Mining and Knowledge Discovery, 7th European Conference, PKDD 2003*.

Hotho, A., Staab, S., & Stumme, G. (2003b). Ontologies improve text document clustering. In *Proc. of the ICDM 03, The 2003 IEEE International Conference on Data Mining*, (pp. 541–544).

Hotho, A., Staab, S., & Stumme, G. (2003c). Wordnet improves Text Document Clustering. In *Proceedings of the Semantic Web Workshop of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, Toronto, Canada. ACM Press.

Ide, N. & Véronis, J. (1998). Introduction to the special issue on word sense disambiguation: The state of the art. *Computational Linguistics*, *24*(1), 1–40.

Joachims, T. (1998). Text Categorization with Support Vector Machines: Learning With Many Relevant Features. In *Proceedings of ECML-98*.

Jouis, C. (1993). *Contribution a la conceptualisation et a la Modelisation des connaissances a partir d'un analyse linguistique de textes. Realisation d'un prototype: le systeme SEEK*. PhD thesis, Universite Paris III - Sorbonne Nouvelle.

Kehagias, A., Petridis, V., Kaburlasos, V. G., & Fragkou, P. (2000). A Comparison of Word- and Sense-Based Text Categorization Using Several Classification Algorithms. *Journal of Intelligent Information Systems*, *21*(3), 227–247.

Kushal Dave, Steve Lawrence, D. M. P. (2003). Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003*, (pp. 519–528). ACM.

Maedche, A., Pekar, V., & Staab, S. (2002). Ontology learning part one - on discovering taxonomic relations from the web. In *Web Intelligence*. Springer.

Maedche, A. & Staab, S. (2000). Discovering conceptual relations from text. In Horn, W. (Ed.), *Proceedings of the 14th European Conference on Artificial Intellignece (ECAI'2000)*.

Maedche, A. & Staab, S. (2002). Measuring similarity between ontologies. In *Proceedings of the European Conference on Knowledge Acquisition and Management (EKAW)*. Springer.

Maedche, A. & Staab, S. (2004). Ontology learning. In S. Staab & R. Studer (Eds.), *Handbook on Ontologies* (pp. 173–189). Springer.

Missikoff, M., Navigli, R., & Velardi, P. (2002). The usable ontology: An environment for building and assessing a domain ontology. In *Proceedings of the International Semantic Web Conference (ISWC)*.

Pereira, F., Tishby, N., & Lee, L. (1993). Distributional clustering of english words. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, (pp. 183–190).

Poesio, M., Ishikawa, T., im Walde, S. S., & Viera, R. (2002). Acquiring lexical knowledge for anaphora resolution. In *Proceedings of the 3rd Conference on Language Resources and Evaluation*.

Poesio, M. & Vieira, R. (1998). A corpus-based investigation of definite description use. *Computational Linguistics*, *24*(2), 183–216.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, *14*(3), 130–137.

Resnik, P. (1997). Selectional preference and sense disambiguation. In *Proceedings of the ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How?*

Salton, G. (1989). *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley.

Schapire, R. E. & Singer, Y. (2000). BoosTexter: A Boosting-based System for Text Categorization. *Machine Learning*, *39*(2/3), 135–168.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*(1), 1–47.

Seguela, P. (2001). *Construction de modeles de connaissances par analyse linguistique de relations lexicales dans les documents techniques*. PhD thesis, Universite de Toulousse.

Staab, S. & Studer, R. (Eds.). (2004). *Handbook on Ontologies*. Springer.

Steinbach, M., Karypis, G., & Kumar, V. (2000). A comparison of document clustering techniques. In *KDD Workshop on Text Mining*.

Velardi, P., Fabriani, P., & Missikoff, M. (2001). Using text processing techniques to automatically enrich a domain ontology. In *Proceedings of the ACM International Conference on Formal Ontology in Information Systems*.

Volz, R., Studer, R., Maedche, A., & Lauser, B. (2003). Pruning-based identification of domain ontologies. *Journal of Universal Computer Science*, *9*(6), 520–529.

Wang, B. B., Mckay, R. I., Abbass, H. A., & Barlow, M. (2003). A comparative study for domain ontology guided feature extraction. In *Proceedings of the 26th Australian Computer Science Conference (ACSC-2003)*, (pp. 69–78)., Adelaide, Australia. Australian Computer Society, Inc.

Wielinga, B. J., Schreiber, A. T., Wielemaker, J., & Sandberg, J. A. C. (2001). From Thesaurus to Ontology. In *Proceedings of the ACM SIGART International Conference on Knowledge Capture*. ACM Press.