Text Clustering Based on Background Knowledge

Andreas Hotho, Steffen Staab, Gerd Stumme

Institute of Applied Informatics and Formal Description Methods AIFB,
University of Karlsruhe, D–76128 Karlsruhe, Germany
{hotho, staab, stumme}@aifb.uni-karlsruhe.de
http://www.aifb.uni-karlsruhe.de/WBS

Technical Report No. 425

Abstract. Text document clustering plays an important role in providing intuitive navigation and browsing mechanisms by organizing large amounts of information into a small number of meaningful clusters. Standard partitional or agglomerative clustering methods efficiently compute results to this end.

However, the bag of words representation used for these clustering methods is often unsatisfactory as it ignores relationships between important terms that do not co-occur literally. Also, it is mostly left to the user to find out why a particular partitioning has been achieved, because it is only specified extensionally. In order to deal with the two problems, we integrate background knowledge into the process of clustering text documents.

First, we preprocess the texts, enriching their representations by background knowledge provided in a core ontology — in our application Wordnet. Then, we cluster the documents by a partitional algorithm. Our experimental evaluation on Reuters newsfeeds compares clustering results with pre-categorizations of news. In the experiments, improvements of results by background knowledge compared to the baseline can be shown for many interesting tasks.

Second, the clustering partitions the large number of documents to a relatively small number of clusters, which may then be analyzed by *conceptual clustering*. In our approach, we applied Formal Concept Analysis. Conceptual clustering techniques are known to be too slow for directly clustering several hundreds of documents, but they give an intensional account of cluster results. They allow for a concise description of commonalities and distinctions of different clusters. With background knowledge they even find abstractions like “food” (vs. specializations like “beef” or “corn”). Thus, in our approach, partitional clustering reduces first the size of the problem such that it becomes tractable for conceptual clustering, which then facilitates the understanding of the results.

Table of Contents

1 Introduction

With the abundance of text documents available through corporate document management systems and the World Wide Web, the dynamic partitioning of texts into previously unseen categories is a major topic for applications such as information retrieval from databases, business intelligence solutions or enterprise portals.

However, in spite of a long tradition of research in similarity-based text document retrieval [21] there is no clustering method that could function as a panacea to this end. We conjecture the reason to stem from the fact that text document clustering must *simultaneously* deal with quite a number of problems:

1. *Problem of efficiency:* Text document clustering must be efficient because it should be able to do clustering on ad-hoc collections of documents, *e.g.* ones found by a search engine through keyword search.
2. *Problem of effectiveness:* Text document clustering must be effective, i. e., it should relate documents that talk about the same *or a similar* domain. Currently, similarity may only be detected on the basis of correlation — which is not always given when changing from one to a similar domain.
3. *Problem of explanatory power:* Text document clustering should be able to explain to the user why a particular result was constructed, or at least provide him with an intuition. Lack of understandability may pose a much bigger threat to the success of an application that employs text document clustering than a few percentage points decrease in accuracy.
4. *Problem of user interaction and subjectivity:* Applications that employ text document clustering must be able to involve the user. The results should be explained, of course, but it should also be possible to re-focus one's attention on particularly relevant subjects. For instance, a search for "health" might turn up food-related issues that a user might want to explore in details relevant for him, such as "meat", "pork", "beef" and others.

In this paper we explore an original combination of technologies in order to achieve progress on these problems:

1. We base our principal clustering effort on well-known efficient and effective partitional algorithms. More specifically, we use Bi-Section-KMeans, which has been shown to perform as good as other text clustering algorithms — and frequently better (cf. the very seminal paper [24]).
2. We add background knowledge from a general resource, Wordnet, into the text document representation in order to relate similar terms such as "beef", "pork" and "meat". Our experiments demonstrate that the best strategies that involve such background knowledge are never worse than the baseline, but often times better. We will see that the best strategies include word sense disambiguation and feature weighting.
3. The background knowledge is crucial at this point, because it adds explanatory power to the representation of the cluster and, thus, to the input of Conceptual Clustering: Formal Concept Analysis then takes advantage of the semantic relationships between previously isolated terms (synonymy and hypernymy) in order to provide the user with a better explanation of cluster results.
4. We use visualization methods from the field of Formal Concept Analysis in order to let the user navigate and explore the clustering results. Subsequently, we will describe how this navigation works such that the user may find relevant information.

In the remainder of the paper we proceed according to the stream of data going from text documents to a baseline representation (Section 2). The baseline representation is extended to take background knowledge into account (Section 3). The two representations are then clustered by Bi-Section-KMeans and the corresponding performances are compared along different data sets and parameter dimensions (Section 4). Section 5 elaborates the conceptual clustering phase that builds on the results of the partitional clustering with background knowledge, before we discuss the results in Section 6.

2 Preprocessing: Baseline Text Document Representation

For the preprocessing of the documents, we used the text mining system developed at AIFB within the Karlsruhe Ontology Framework KAON.[1] For the clustering experiments described subsequently, we had to prepare different representations of text documents suitable for the clustering algorithms.

Let us first consider documents to be bags of terms. Let $\text{tf}(d,t)$ be the absolute frequency of term $t \in T$ in document $d \in D$, where D is the set of documents and $T = \{t_1, \ldots, t_m\}$ is the set all different terms occurring in D. We denote the term vectors $\vec{t_d} = (\text{tf}(d,t_1), \ldots, \text{tf}(d,t_m))$. In the sequel, we will apply tf also on sets of terms: for $T' \subseteq T$, we let $\text{tf}(d,T') := \sum_{t \in T'} \text{tf}(d,t)$.

As initial approach we have produced this standard representation of the texts by term vectors (cf. [22]). As a slightly more advanced approach, we have taken into account several combinations of stopword removal, stemming of terms, pruning of terms that appear infrequently, and weighting by tfidf, which modify the term vectors $\vec{t_d}$ accordingly.

In the sequel, we will need the notion of the centroid of a set X of term vectors. It is defined as the mean value $\vec{t_X} := \frac{1}{|X|} \sum_{\vec{t_d} \in X} \vec{t_d}$ of its term vectors.

2.1 Stopword Removal

Stopwords are words which are considered as non–descriptive within a bag–of–words approach. They typically comprise prepositions, articles, etc. Following common practice, we removed stopwords from T, using a standard list with 571 stopwords.[2]

2.2 Stemming

We have processed our text documents using the Porter stemmer introduced in [20]. Instead of using the original terms in the documents, we have computed the frequency of stemmed terms (modifying T correspondingly) and used them to construct a vector representation $\vec{t_d}$ for each text document. The length of the resulting vectors is given by the number of different stemmed terms in the text corpus.

2.3 Pruning

For some empirical investigations we have entirely discarded all terms appearing rarely. For a pre-defined threshold δ, a term t is discarded from the representation (i. e., from the

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

[2] http://www.aifb.uni-karlsruhe.de/WBS/aho/clustering

set T), if $\sum_{d \in D} \mathrm{tf}(d,t) \leq \delta$. We have used the values 0, 5 and 30 for δ. The rationale behind pruning is that infrequent terms do not help with identifying appropriate clusters, but they may still add noise to the distance measures degrading overall performance. [3]

2.4 TFIDF Weighting

Tfidf weighs the frequency of a term 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 balance between the two extremes and, hence, are expected to be better able to contribute to clustering results.

Definition: The tfidf $(\text{term frequency}-\text{inverted document frequency})$ [4] of term t in document d is defined by:

$$\mathrm{tfidf}(d,t) := \log(\mathrm{tf}(d,t)+1) * \log\left(\frac{|D|}{\mathrm{df}(t)}\right)$$

where $\mathrm{df}(t)$ 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 $\vec{t_d} := (\mathrm{tf}(d,t_1),\ldots,\mathrm{tf}(d,t_m))$ by $\vec{t_d} := (\mathrm{tfidf}(d,t_1),\ldots,\mathrm{tfidf}(d,t_m))$.

There are more sophisticated measures than tfidf in the literature (see, e. g., [2]), but we abstract herefrom, as this is not the main topic of our approach.

2.5 Combination of the Preprocessing Steps

Based on the initial text document representation, we have first applied stopword removal. Then we performed stemming, pruning and tfidf weighting in all different combinations. This also holds for the initial document representation involving background knowledge described subsequently. When stemming and/or pruning and/or tfidf weighting was performed, we have always performed them in the order in which they have been listed here.

3 Compiling Background Knowledge into the Text Document Representation

The background knowledge we have exploited is given through a simple ontology. We first describe its structure, then the actual ontology and the integration into the initial text document representation through various strategies. Like the preprocessing strategies described before, the different strategies for compiling background knowledge into the text document representations may be arbitrarily combined, and will modify the term vectors accordingly.

[3] We investigated also the influence of the document frequency of a term t (cf. Section 2.4) for pruning, but it showed that this parameter hardly effects the clustering results.

[4] In the literature, different authors have used the term “tfidf” for different weighting schemes.

3.1 Ontology

The background knowledge we will exploit further on is encoded in a *core ontology*. We here present those parts of our wider ontology definition (cf. [3]) that we have exploited:

Definition: A *core ontology* is a tuple $\mathcal{O} := (C, \leq_C)$ consisting of a set C whose elements are called *concept identifiers*, and a partial order $\leq_C$ on C, called *concept hierarchy* or *taxonomy*.

Often we will call concept identifiers just *concepts*, for sake of simplicity.

Definition: If $c_1 <_C c_2$, for $c_1, c_2 \in C$, then c_1 is a *subconcept of* c_2, and c_2 is a *superconcept of* c_1. If $c_1 <_C c_2$ and there is 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. We note this by $c_1 \prec c_2$.

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: 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*, 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, we define, for $s \in S_C$, $Ref_C(s) := \{c \in C \mid (s, c) \in Ref_C\}$ and, for $c \in C$, $Ref_C^{-1}(c) := \{s \in S_C \mid (s, c) \in Ref_C\}$.

An *ontology with lexicon* is a pair $(\mathcal{O}, Lex)$ where $\mathcal{O}$ is an ontology and Lex is a lexicon for $\mathcal{O}$.

This definition allows for a very generic approach towards using ontologies for clustering. For the purpose of actual evaluation of clustering with background knowledge, we needed a specific resource, which is large and general enough, while we wanted to avoid expensive modeling. Therefore, we have chosen Wordnet 1.7.[5] Wordnet [15] comprises a core ontology and a lexicon. It consists of 109377 concepts (synsets in Wordnet terminology) and 144684 lexical entries[6] (called words in Wordnet). One example synset is "foot, ft" and a corresponding word is "foot". In Wordnet, the function Ref_C relates terms if they have a lexical entry (e.g., s_1 = "foot" and s_2 = "feet") with their corresponding concepts (e.g., synsets c_1 = "foot, ft", c_2 = "foot, human foot, pes", ...). Thus, for a term t appearing in a document d, $Ref_C(t)$ allows for retrieving its corresponding concepts.

In addition, Wordnet provides a ranking on the set $Ref_C(s)$ for each lexical entry s indicating the frequency of its usage in English language. For example, $Ref_C(s_1)$ returns as the first concept c_1 and then c_2. Corresponding to our definition of a core ontology, Wordnet also offers access functions to its concept hierarchy $\leq_C$.

So far, from all the descriptions given in Wordnet, we have exploited only information about nouns. I.e., we have used only 68.1% of the synsets available in Wordnet.

Using the morphological capabilities of Wordnet rather than a Porter stemmer we achieved improved results. Therefore, when using background knowledge, the Porter stemmer has only been applied on terms that do not appear as lexical entries in Wordnet.

[5] freely available from http://www.cogsci.princeton.edu/˜wn/obtain.shtml

[6] The actual number of lexical entries is higher in our count, as for one stem like "foot", Wordnet includes several morphological derivations like "feet".

3.2 Strategies: Add Concepts / Replace Terms by Concepts / Concept Vector Only

Enriching the term vectors with concepts from the core ontology has two benefits. First it resolves synonyms; and second it introduces more general concepts which help identifying related topics. For instance, a document about beef may not be related to a document about pork by the cluster algorithm if there are only 'beef' and 'pork' in the term vector. But if the more general concept 'meat' is added to both documents, their semantical relationship is revealed. We have investigated different strategies for adding or replacing terms by concepts:

Add Concepts ("add"[7]). When applying this strategy, we have extended each term vector $\vec{t_d}$ by new entries for Wordnet concepts c appearing in the document set. Thus, the vector $\vec{t_d}$ was replaced by the concatenation of $\vec{t_d}$ and $\vec{c_d}$, where $\vec{c_d} := (\text{cf}(d, c_1), \ldots, \text{cf}(d, c_l))$ is the concept vector with $l = |C|$ and $\text{cf}(d, c)$ denotes the frequency that a concept $c \in C$ appears in a document d as indicated by applying the reference function Ref_C to all terms in the document d. For a detailed definition of cf, see next subsection.
Hence, a term that also appeared in Wordnet as a synset would be accounted for at least twice in the new vector representation, i. e., once as a part of the old $\vec{t_d}$ and at least once as a part of $\vec{c_d}$. It could be accounted for also more often, because a term like "bank" has several corresponding concepts in Wordnet.

Replace Terms by Concepts ("repl"). This strategy works like "Add Concepts" but it expels all terms from the vector representations $\vec{t_d}$ for which at least one corresponding concept exists. Thus, terms that appear in Wordnet are only accounted at the concept level, but terms that do not appear in Wordnet are not discarded.

Concept Vector Only ("only"). This strategy works like "Replace Terms by Concepts" but it expels *all* term frequencies from the vector representations. Thus, terms that do not appear in Wordnet are discarded. $\vec{c_d}$ is used to represent the documents d.

3.3 Strategies for Disambiguation

The assignment of terms to concepts in Wordnet is ambiguous. Therefore, adding or replacing terms by concepts may add noise to the representation and may induce a loss of information. Therefore, we have also investigated how the choice of a "most appropriate" concept from the set of alternatives may influence the clustering results.

While there is a whole field of research dedicated to word sense disambiguation (e.g., cf. [11]), it has not been our intention to determine which one could be the most appropriate, but simply whether word sense disambiguation is needed at all. For this purpose, we have considered two simple disambiguation strategies besides of the baseline:

All Concepts ("all"). The baseline strategy is not to do anything about disambiguation and consider all concepts for augmenting the text document representation. Then, the concept frequencies are calculated as follows:

$$\text{cf}(d, c) := \text{tf}(d, \{t \in T \mid c \in Ref_C(t)\}) \ .$$

[7] These abbreviations are used below in Section 4.4

First Concept ("first"). As mentioned in Sec. 3.1 Wordnet returns an *ordered* list of concepts when applying Ref_C to a set of terms. Thereby, the ordering is supposed to reflect how common it is that a term reflects a concept in "standard" English language. More common term meanings are listed before less common ones. Completely ignoring the context of a term, the probability that the first concept returned for the term was in the mind of the writer is supposed to be maximized.
For a term t appearing in S_C, this strategy counts only the concept frequency cf for the first ranked element of $\mathit{Ref}_C(t)$, i.e. the most common meaning of t. For the other elements of $\mathit{Ref}_C(t)$, frequencies of concepts are not increased by the occurrence of t. Thus the concept frequency is calculated as follows:

$$\text{cf}(d, c) := \text{tf}(d, \{t \in T \mid \text{first}(\mathit{Ref}_C(t)) = c\})$$

where first(Ref_C) gives the first concept $c \in \mathit{Ref}_C$ according to the order from Wordnet.

Disambiguation by Context ("context"). The sense of a term t that refers to several different concepts $\mathit{Ref}_C(t) := \{b, c, \ldots\}$ may be disambiguated by the following simple strategy[8]:

1. Define the semantic vicinity of a concept c to be the set of all its direct sub- and superconcepts $V(c) := \{b \in C | c \prec b \text{ or } b \prec c\}$.
2. Collect all terms that could express a concept from the conceptual vicinity of c by $U(c) := \bigcup_{b \in V(c)} \mathit{Ref}_C^{-1}(b)$.
3. The function dis: $D \times T \to C$ with dis$(d, t) :=$ first$\{c \in \mathit{Ref}_C(t) \mid c$ maximizes tf$(d, U(c))\}$ disambiguates term t based on the context provided by document d.
4. Let cf$(d, c) :=$ tf$(d, \{t \in T \mid \text{dis}(d, t) = c\})$.

3.4 Strategies for considering Hypernyms

The third set of strategies varies the amount of background knowledge. Its principal idea is that if a term like 'beef' appears, one does not only represent the document by the concept corresponding to 'beef', but also by the concepts corresponding to 'meat' and 'food' etc. up to a certain level of generality. The following procedure realizes this idea by adding to the concept frequency of higher level concepts in a document d the frequencies that their subconcepts (at most r levels down in the hierarchy) appear, *i.e.* for $r \in \mathbb{N}$:

The vectors we consider are of the form

$$\vec{t_d} := (\text{tf}(d, t_1), \ldots, \text{tf}(d, t_m), \text{cf}(d, c_1), \ldots, \text{cf}(d, c_n))$$

(the concatenation of an initial term representation with a concept vector). Then the frequencies of the concept vector part are updated in the following way: For all $c \in C$, replace cf(d, c) by

$$\text{cf}'(d, c) := \sum_{b \in H(c, r)} \text{cf}(d, b) \quad ,$$

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 subconcceps in the taxonomy. In particular $H(c, \infty)$ returns all subconcepts of c. For the following parameters this implies:

[8] This strategy is a simplified version of [1].

$r = 0$: The strategy does not change the given concept frequencies.
$r = n$: the strategy adds to each concept the frequency counts of all subconcepts in the n levels below it in the ontology.
$r = \infty$: The strategy adds to each concept the frequency counts of all its subconcepts.

4 Partitional Clustering

On the text document representations resulting out of different combinations of preprocessing strategies we have run Bi-Section-KMeans (cf. [24]) using the cosine-distance on the different vector representations.

4.1 Cosine-distance

We calculate the similarity between two text documents $d_1, d_2 \in D$ by computing the cosine of the angle between the vectors $\vec{t}_1, \vec{t}_2$ representing them:

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

4.2 Bi-Section-KMeans

For the clustering, we looked for a fast and good quality clustering algorithm, which would also be able to deal with the large size of the Reuters dataset. In [24] it was shown that Bi-Section-KMeans is a fast and high-quality clustering algorithm for text documents which is frequently outperforming standard KMeans as well as agglomerative clustering techniques.

Bi-Section-KMeans is based on the KMeans algorithm. It repeatedly splits the largest cluster (using KMeans) until the desired number of clusters is obtained:

```
Input: The number $k$ of desired clusters.
Output: A partitioning $\mathbb{P}$ of the set $D$ of documents (i. e., a set $\mathbb{P}$ of $k$ disjoint
subsets of $D$ with $\bigcup_{P \in \mathbb{P}} P = D$).
(1) Let $\mathbb{P} := \{D\}$.
(2) For $i := 1$ to $k - 1$ do
        • Select $P \in \mathbb{P}$ with maximal cardinality.
        • Choose randomly two data points in $P$ as starting centroids $\vec{t}_{P_1}$ and $\vec{t}_{P_2}$.
        • Assign each point of $P$ to the closest centroid, splitting thus $P$ in two
          clusters $P_1$ and $P_2$.
        • (Re-)calculate the cluster centroids $\vec{t}_{P_1}$ and $\vec{t}_{P_2}$ of $P_1$ and $P_2$.
        • Repeat the last two steps until the centroids do not change anymore.
        • Let $\mathbb{P} := (\mathbb{P} \setminus \{P\}) \cup \{P_1, P_2\}$.
```

4.3 Evaluation Setting

This section describes the setting in which the experiments have been performed. The principal idea behind the experiments was the comparison of clustering results on a standard text corpus against a manually predefined categorization of the corpus. Such a predefined categorization exists only for few text corpora.

We have chosen the Reuters-21578 news corpus (cf. section 4.3), because it comprises an *a priori* categorization of documents, its domain is broad enough to be realistic, and the content of the news were understandable for non-experts (like us) in order to be able to explain results. Furthermore, Reuters-21578 is a well-known, freely available and well investigated corpus allowing even for future comparisons only based on numbers instead of direct experiments.

Important reasons for us to use Wordnet as a core ontology in conjunction with Reuters-21578 as a corpus were that Wordnet is freely available and that it has not been specifically designed to facilitate the clustering task. We expect further improvements when ontologies specifically designed for some concrete task can be used.

In the experiments we have varied the different strategies for plain term vector representation and for vector representations containing background knowledge as elaborated in Sections 2 and 3. We have clustered the representations using Bi-Section-KMeans and have compared the pre-categorization with our clustering results using standard measures for this task, i. e., purity and inverse purity (defined below).

Evaluation Measure: Purity and Inverse Purity Purity is based on the precision measure as well-known from information retrieval (cf. [19]). Each resulting cluster P from a partitioning $\mathbb{P}$ of the overall document set D is treated as if it were the result of a query. Each set L of documents of a partitioning $\mathbb{L}$ which is obtained by manually labeling is treated as if it were the desired set of documents for a query. The two partitionings $\mathbb{P}$ and $\mathbb{L}$ are then compared as follows.

The precision of a cluster $P \in \mathbb{P}$ for a given category $L \in \mathbb{L}$ is given by

$$\text{Precision}(P, L) := \frac{|P \cap L|}{|P|} \tag{1}$$

The overall value for purity is computed by taking the weighted average of maximal precision values:

$$\text{Purity}(\mathbb{P}, \mathbb{L}) := \sum_{P \in \mathbb{P}} \frac{|P|}{|D|} \max_{L \in \mathbb{L}} \text{Precision}(P, L). \tag{2}$$

For some selected parameter combinations that proved to be very good wrt. purity, we also investigated their InversePurity:

$$\text{Inverse Purity}(\mathbb{P}, \mathbb{L}) := \sum_{L \in \mathbb{L}} \frac{|L|}{|D|} \max_{P \in \mathbb{P}} \text{Precision}(L, P). \tag{3}$$

Both measures have the interval $[0, 1]$ as range. Their difference is that purity measures the purity of the resulting clusters when evaluated against a pre-categorization, while inverse purity measures how pure the pre-defined categories are when split up into clusters. Thus, purity achieves an “optimal” value of 1 when k equals $|D|$, whereas inverse purity achieves an “optimal” value of 1 when k equals 1. Another name in the literature for inverse purity is microaveraged precision. The reader may note that, in our scenario, microaveraged precision is identical to microaveraged recall (cf. e.g. [23]).

The Reuters-Corpus We have performed all evaluations on the Reuters-21578 document set [13][9] or on parts of it. In order to be able to perform comparisons with *a priori* categorizations, we have restricted ourselves to the 12344 documents that were manually classified by Reuters. Documents in the manually classified set were labeled with zero, one, or more of the 135 pre-defined categories. [10]

The lack of a label indicates that the human annotator could not find an adequate category. We gathered all the documents without any category label into a new category “defnoclass”.[11]

Standard measures like purity (or mutual information or entropy) only allow for the comparison of two partitionings, but they do not allow for the comparison of structures when documents are manually assigned to *several* categorizations and/or documents are automatically assigned to *multiple* clusters. Therefore, we have only selected the first label of each document retaining a manual categorization of documents into overall 82 categories, including “defnoclass”.

This way, we end up with a preprocessed version of the Reuters-21578 corpus, which we call PRC. It consists of 12344 documents partitioned into 82 Reuters categories.

Derived Corpora Even with the preprocessing, the resulting corpus exhibits several problems for evaluation purposes:

1. Most documents are assigned to one category out of a small subset of all categories (cf. Figure 1).[12] For this situation the purity measure typically indicates very good results, as medium sized clusters — as typically produced by Bi-Section-KMeans— are wholly contained in one of the few large categories. Therefore, improvements for purity are hard to produce, and even when they occur they are hard to observe with the purity measure.
2. There are a few categories which contain very few documents. Some categories only contain one document.
3. As described above, we used only the first label assigned by the Reuters domain experts for evaluation, as the purity measure does not allow for multiple labels. This implies that our classification may fail according to that measure, even though the assignment would have been correct according to the second (or third) label. As mentioned before, when using a clustering algorithm allowing for multiple assignments, we would not be able to perform an evaluation with a generally accepted evaluation measure like purity.

While we must live with the third problem (unless we want to use an ideosyncratic, specifically developed evaluation measure), we have dealt with the first two problems by modifying the number of documents available in a category. This was done by specifying a minimum number of documents per category. Below the threshold, e.g. 5 or 25, the entire category and its documents would be discarded from PRC in order to create a new corpus, e.g. PRC-min5 and PRC-min25, respectively. Also, we have specified a maximum number

[9] http://www.daviddlewis.com/resources/testcollections/reuters21578/

[10] The categories are called “topics” in Reuters-21578. To be more general, we will refer to them as “category” in the sequel.

[11] The 12344 documents are indicated by an attribute “TOPIC” set to yes and contain the text surrounded by the “BODY” tag.

[12] For instance, the largest category contains 3760 documents.

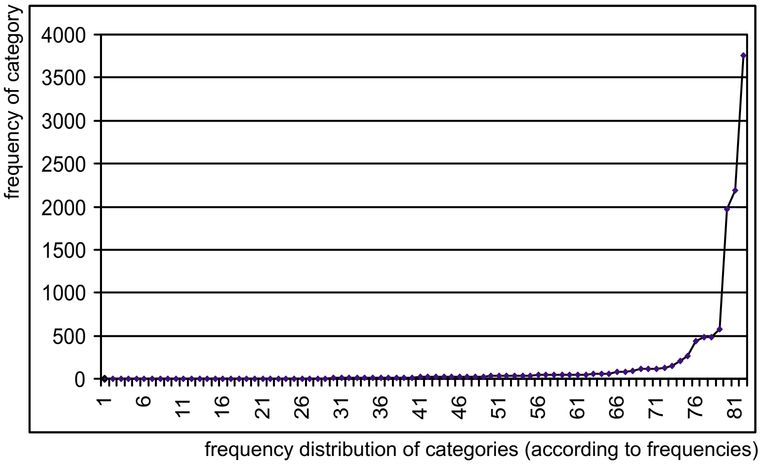

Fig. 1. Distribution of documents to categories for PRC.

of documents per category. Above the threshold, e.g. 100, we would retain the category, but select an arbitrary sample of, e.g. 100, documents, thus constructing a new corpus PRC-max100. We have also investigated combined constraints, i. e., derived corpora like PRC-min25-max100.

Overall we have derived five additional corpora:

- PRC-max20: This corpus contains only categories with very few documents (max 20). Thereby, PRC-max20 very nicely shows the effects of background knowledge on categories with extremely few documents. It consists of 82 categories with an average of 12.62 documents per category (standard deviation of 8.18).
- PRC-min15-max20: This is a very homogeneous corpus (almost uniform distribution of documents to categories). All 46 categories contain 15 to 20 documents (average of 19.54, standard deviation of 1.15).
- PRC-max100: The corpus consists of 82 categories, which exhibit a less uniform distribution of documents to categories, but which do not flood the evaluation with overly large categories (average of 33.59 documents per category with a standard deviation of 36.28).
- PRC-min15-max100: This corpus is like PRC-max100, but categories with extremely few documents are discarded — thus, “outlier categories” may be ignored in the evaluation (cf. Figure 2). Thus, PRC-min15-max100 consists of 46 categories with an average of 56.93 documents (standard deviation of 33.12).
- PRC-min15: This one is like PRC, but it consists of 46 categories eliminating outliers. The average number of documents per category is 672.7 and the standard deviation is 265.39.
- PRC: For a direct comparison enclosed here — PRC contains 12344 documents, average 150.54 documents, standard deviation 520.3 (cf. Figure 1).

4.4 Results

This section describes the combination of parameter values for which tests have been performed and highlights some of them.

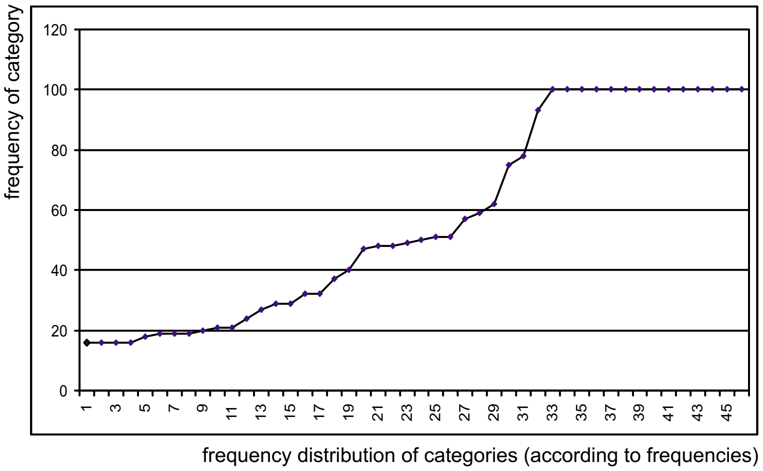

Fig. 2. Distribution of documents to categories for PRC-min15-max100.

Table 1. List of all parameters

parameter name	used values
corpus	PRC, PRC-min15, PRC-max100, PRC-min15-max100, PRC-max20, PRC-min15-max20
stopword removal	yes
stemming	applied only without background knowledge
word pruning	no, 5 words, 30 words
weighting of the term vector	tfidf, no weighting
integration of background knowledge	add, replace, only
amount of hypernyms	0 and 5
words sense disambiguation	without, first, context
cluster count k	5,10,20,30,50,60,70,100

General Procedure Each evaluation result described in the following denotes an average from 20 test runs performed on a given corpus for a given combination of parameter values with randomly chosen initial values for Bi-Section-KMeans.

Table 1 summarizes the different dimensions that we investigated and that led us to an investigation of $20 \times 8 \times 6 \times 2 \times 3 = 5760$ clustering experiments without background knowledge (number of test runs $\times$ number of different cluster counts k $\times$ number of different corpora $\times$ number of weighting schemes $\times$ number of pruning strategies applied) and $20 \times 8 \times 6 \times 2 \times 3 \times 3 \times 3 \times 2 = 103680$ clustering experiments with background knowledge (number of test runs $\times$ number of different cluster counts k $\times$ number of different corpora $\times$ number of weighting schemes $\times$ number of pruning strategies applied $\times$ number of strategies for applying background knowledge $\times$ number of word sense disambiguation strategies $\times$ number of different amounts of hypernyms).

We varied the number of clusters k to be computed by Bi-Section-KMeans from $k := 5, 10, 20, 30, 50, 60, 70$ to 100. Purity values 'improve' with increased number of clusters — attaining 1.0 in the unrealistic case when k is set to equal the number of documents. Given that even human annotators are far from agreeing on any particular labeling, we did not and could not expect to find perfect matches of document clusters to categories *for reasonable values* of k.

In all the test runs we have used stopword removal (Section 2.1). Stemming (Section 2.2) was applied only when we did not make use of background knowledge. We have performed tests with and without using pruning (Section 2.3). We set the different thresholds to 0, 5, and 30. Furthermore, we have varied the strategies for weighting from "no weighting" to tfidf weighting (Section 2.4).

We have performed all combinations of these parameters on the six text corpora presented in Section 11. We did the tests using representations without (described in Section 13) and with background knowledge (Section 13). The text document representations consisted of term vectors of length 544 to 20574 and concept vectors (or mixed term/concept vectors) of length 2627 to 36322, respectively. We have selected some results for presentation in the following.[13]

Clustering without Background Knowledge

Effects of tfidf Weighting. We have observed that tfidf weighting decisively increased purity values irrespective of what the combination of parameter values was (see for instance Table 2).

Effects of Pruning. Pruning with a threshold of 5 or 30 has not always shown an effect. But it increased purity values when it was combined with tfidf weighting and applied to corpora with few documents per category. To elucidate the latter: Clustering of PRC-min15-max20 with $k = 100$ (alternatively: $k = 50$) increased the purity from 49,8% (alt.: 42,5%) without pruning to 60% (alt.: 52,1%) when pruning with a threshold of 30.

For corpora with large numbers of documents per category, e.g. PRC-min15, the corresponding difference has become almost negligible, namely 0.6%. Experiments with a threshold above 30 showed no further improvement on the PRC-min15 corpus.

Clustering with Background Knowledge For clustering using background knowledge, we also performed pruning and tfidf weighting as described just before. The thresholds and modifications have been enacted on concept frequencies (or mixed term/concept frequencies) instead of term frequencies only. We have computed the purity results for varying parameter combinations as described before.

Evaluation on PRC-min15-max100. A subset of all cross evaluations is depicted in Figure 3 and Table 2. Each data point is the average over 20 runs of Bi-Section-KMeans and indicates a combination of values as follows:

X-axis: On the X-axis, different parameter combinations are indicated. From bottom to top there are:

- Without background knowledge (Section 2) vs. with background knowledge (Section 3), (Ontology = false/true).
- No use of hypernyms (r=0) vs. five levels of hypernyms added to concept frequencies ($r = 5$), cf. Section 3.4 (Hypdepth = 0 / 5).

[13] The complete data set can be found at http://www.aifb.uni-karlsruhe.de/WBS/aho/clustering.

– Disambiguation strategy: All concepts / First concept / Disambiguation by context; cf. Section 3.3 (Hypdis = All/First/Context).
– Add Concepts vs. Replace Terms by Concepts vs. Concept Vector Only; cf. Section 3.2 (Hypint = add/repl/only).

Y-axis: On the Y-axis the resulting purity averaged over 20 test runs for each data point (as defined in Section 4.3) is shown.

Different Lines represent different combinations of tfidf weighting vs. no weighting with different pruning thresholds (0 vs. 5 vs. 30).

Results. The baseline, i. e., the representation without background knowledge, in Figure 3 is given by the best value, 57%, in the leftmost sector (the one for tfidf weighting and a pruning threshold of 30). The best overall value is achieved by the following combination of strategies: Background knowledge with five levels of hypernyms ($r = 5$), using "disambiguation by context" and term vectors extended by concept frequencies. Purity values then reached 61,8%, thus yielding a relative improvement of 8.4% compared to the baseline (cf. Table 2).

Without the application of tfidf weighting, all different parameter combinations achieve lower values. Also the difference between the best baseline result (47%) and the best results achieved by adding background knowledge (48,6%) decreases considerably. Furthermore, strategies that consider hypernyms without weighting, like $r = 5$ without tfidf weighting, even decrease the purity compared to the baseline.

Varying the number of clusters k for the parameter combinations described in Figure 3 has hardly altered the overall picture. The reported results for dataset PRC-min15-max100 are very similar to the results of PRC-max100.

Significance. We have applied a T-test to check for the significance with a confidence of 99.5%. Unless stated explicitly otherwise, all differences that are mentioned are significant within the confidence interval $\alpha = 0.5\%$.

Evaluation on the Complete PRC. Figure 4 describes the result like in Figure 3, but this time applying the parameter combinations on the complete corpus PRC. Overall, similar qualitative results are achieved on PRC as on PRC-min15-max100, but the margin between the best baseline (75.1%) and the best clustering result with background knowledge (75.4% for using no hypernyms, disambiguating by context and adding concepts) becomes almost negligible. The difference is not significant with $\alpha = 0.5\%$.

In order to investigate why the improvement by background knowledge became so small for PRC, we continued with two further investigations.

*Inverse Purity.*As may be seen from the description in Section 4.3, purity overly favors large split counts and does not discount evaluation results when splitting up large categories. Therefore, we have investigated how the inverse purity values would be affected for the best baseline (in terms of purity) and a typically good strategy based on background knowledge (again measured in terms of purity). Tables 3 and 4 summarize the results.

As seen before, the purity values for PRC do not differ significantly between our typically good strategies based on background knowledge (Hypdis = context, prune=30,

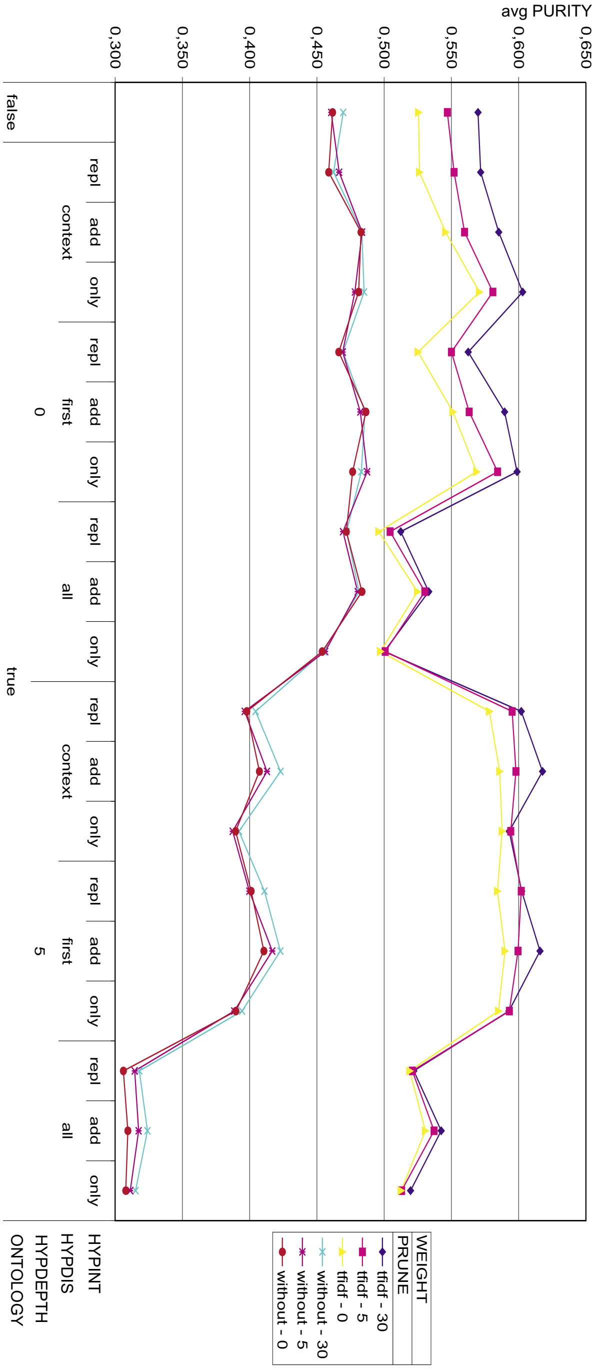

Fig. 3. Comparing clustering without background knowledge (leftmost column) against various combinations of parameter settings using background knowledge on PRC-min15-max100 with $k = 60$.

Table 2. Using Purity to compare clustering without background knowledge (first row) against various combinations of parameter settings using background knowledge on PRC-min15-max100 with $k = 60$ (avg denotes average over 20 cluster runs and std denotes standard deviation).

				WEIGHT PRUNE					
				tfidf			without		
				30	5	0	30	5	0
ONTO	HYPDEPTH	HYPDIS	HYPINT	Purity	Purity	Purity	Purity	Purity	Purity
				avg ± std	avg ± std	avg ± std	avg ± std	avg ± std	avg ± std
false				$\boxed{0,57 \pm 0,019}$	0,547 ± 0,018	0,526 ± 0,023	0,47 ± 0,012	0,461 ± 0,015	0,462 ± 0,015
true	0	context	repl	0,572 ± 0,011	0,552 ± 0,02	0,526 ± 0,016	0,462 ± 0,015	0,467 ± 0,01	0,459 ± 0,016
			add	0,585 ± 0,014	0,56 ± 0,019	0,546 ± 0,019	0,483 ± 0,013	0,484 ± 0,011	0,483 ± 0,014
			only	0,603 ± 0,019	0,581 ± 0,017	0,571 ± 0,022	0,485 ± 0,01	0,478 ± 0,012	0,481 ± 0,009
		first	repl	0,562 ± 0,014	0,55 ± 0,014	0,525 ± 0,018	0,469 ± 0,011	0,469 ± 0,016	0,466 ± 0,019
			add	0,59 ± 0,015	0,563 ± 0,015	0,551 ± 0,016	0,486 ± 0,017	0,482 ± 0,015	0,486 ± 0,015
			only	0,599 ± 0,016	0,584 ± 0,019	0,569 ± 0,017	0,483 ± 0,01	0,487 ± 0,01	0,477 ± 0,017
		all	repl	0,512 ± 0,015	0,504 ± 0,015	0,496 ± 0,015	0,472 ± 0,014	0,469 ± 0,01	0,472 ± 0,011
			add	0,533 ± 0,014	0,53 ± 0,013	0,525 ± 0,016	0,481 ± 0,011	0,48 ± 0,016	0,484 ± 0,013
			only	0,5 ± 0,016	0,501 ± 0,013	0,497 ± 0,02	0,455 ± 0,009	0,456 ± 0,011	0,454 ± 0,011
	5	context	repl	0,602 ± 0,015	0,595 ± 0,017	0,578 ± 0,017	0,404 ± 0,009	0,396 ± 0,013	0,398 ± 0,013
			add	$\boxed{0,618 \pm 0,015}$	0,598 ± 0,015	0,586 ± 0,017	0,423 ± 0,013	0,413 ± 0,015	0,407 ± 0,016
			only	0,593 ± 0,01	0,594 ± 0,011	0,588 ± 0,013	0,392 ± 0,008	0,387 ± 0,014	0,39 ± 0,012
		first	repl	0,602 ± 0,014	0,602 ± 0,015	0,584 ± 0,015	0,411 ± 0,012	0,4 ± 0,014	0,401 ± 0,013
			add	0,616 ± 0,015	0,6 ± 0,017	0,59 ± 0,016	0,423 ± 0,014	0,417 ± 0,011	0,411 ± 0,016
			only	0,593 ± 0,011	0,593 ± 0,016	0,585 ± 0,011	0,394 ± 0,01	0,389 ± 0,009	0,39 ± 0,012
		all	repl	0,522 ± 0,01	0,521 ± 0,015	0,519 ± 0,015	0,318 ± 0,009	0,315 ± 0,01	0,306 ± 0,011
			add	0,542 ± 0,012	0,537 ± 0,017	0,531 ± 0,012	0,324 ± 0,006	0,318 ± 0,01	0,309 ± 0,008
			only	0,52 ± 0,014	0,513 ± 0,017	0,513 ± 0,016	0,315 ± 0,009	0,311 ± 0,006	0,308 ± 0,009

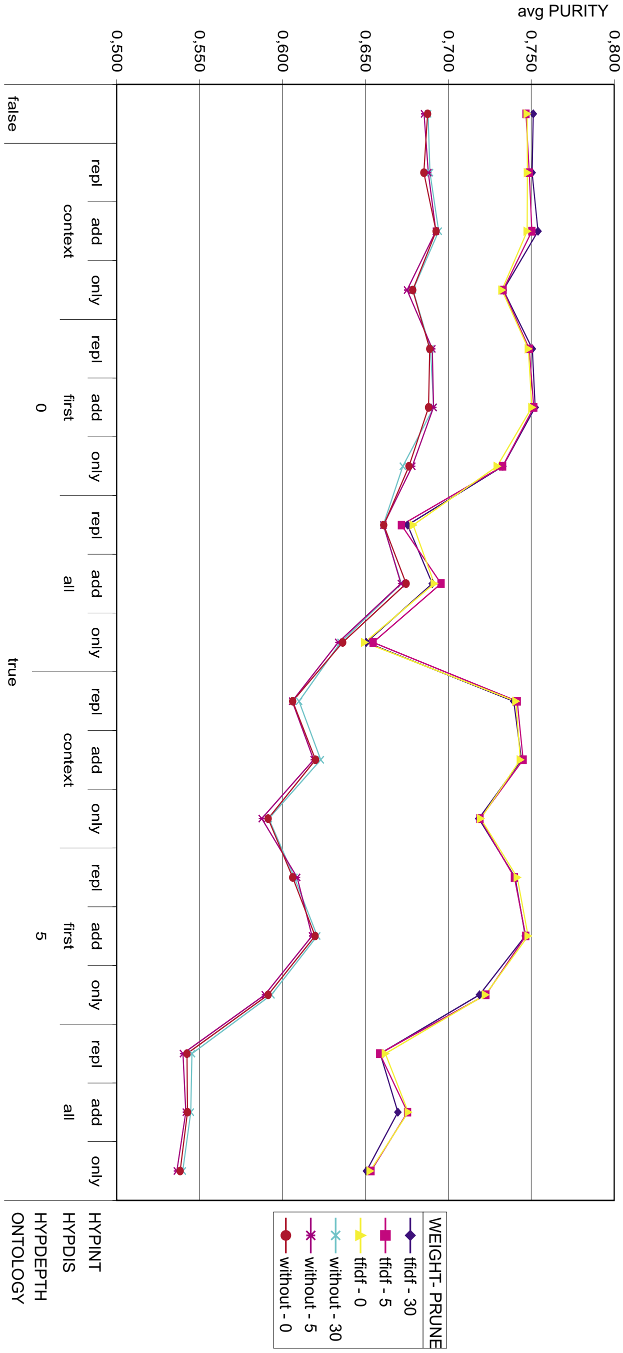

Fig. 4. Comparing clustering without background knowledge (leftmost column) against various combinations of parameter setting using background knowledge on PRC with $k = 60$.

Table 3. Results on PRC $k = 60$, prune=30 (with background knowledge also HYPDIS = context, avg denotes average over 20 cluster runs and std denotes standard deviation)

ONTO	HYPDEPTH	HYPINT	Purity	InversePurity
			avg ± std	avg ± std
false			0,751 ± 0,006	0,263 ± 0,007
true	0	add	0,755 ± 0,007	0,269 ± 0,009
		only	0,736 ± 0,008	0,266 ± 0,009
	5	add	0,746 ± 0,006	0,272 ± 0,007
		only	0,721 ± 0,007	0,271 ± 0,010

HYPDEPTH=5, HYPINT=add) and the best baseline. For inverse purity, the improvement by the same background knowledge strategy is small, but significant within a confidence interval of 0.5%. On PRC-min15-max100, purity *and* inverse purity are clearly improved by the same background knowledge strategy.

Table 4. Results on PRC-min15-max100 $k = 60$, prune=30 (with background knowledge also HYPDIS = context, avg denotes average over 20 cluster runs and std denotes standard deviation)

ONTO	HYPDEPTH	HYPINT	Purity	InversePurity
			avg ± std	avg ± std
false			0,57 ± 0,019	0,479 ± 0,016
true	0	add	0,585 ± 0,014	0,492 ± 0,017
		only	0,603 ± 0,019	0,504 ± 0,021
	5	add	0,618 ± 0,015	0,514 ± 0,019
		only	0,593 ± 0,01	0,500 ± 0,016

Inverse Purity and Variance Analysis. Considering the significant, but nevertheless small improvement of inverse purity on PRC in contrast to the very clear improvements on PRC-min15-max100, we have investigated when and why background knowledge will improve the results of Bi-Section-KMeans. We investigated the within-class variance of the Reuters categorization of PRC. For $X \subseteq D$ the variance is defined as:

$$\mathrm{var}(X) := \sum_{d \in X} ||\vec{t_d} - \vec{t_X}||^2 \quad . \tag{4}$$

Based on this definition, we define the normalized variance within a class L as follows, where the denominator performs a normalization adjusting the variance to the corresponding overall variance of D:

$$\mathrm{var}_{in}(L) := \frac{\mathrm{var}(L)}{\mathrm{var}(D)}. \tag{5}$$

This variance can be computed both for vector representations with or without background knowledge. We thus obtain two values for each class L, namely $\mathrm{var}\,_{in}^{with}(L)$ and $\mathrm{var}_{in}^{without}(L)$.[14]

The normalized difference of the variances is obtained by

$$\mathrm{vd}(L) := \frac{\mathrm{var}_{in}^{with}(L) - \mathrm{var}_{in}^{without}(L)}{\mathrm{var}_{in}^{without}(L)}, \tag{6}$$

The decreasing line in Figure 5 shows this normalized difference of the within-class variance between the representations with (strategy hypdepth=5, hypint=add, hypdis=context, prune=30) and without background knowledge. As becomes evident, for the large majority of pre-defined categories, background knowledge reduces the within-class variance.

Furthermore, there is a clear tendency that the unsupervised reduction of variance within predefined categories improves the inverse purity in comparison to the best baseline. This tendency becomes evident when one compares the variance difference against the individual inverse purity values

$$\mathrm{ipv}(L, \mathbb{P}) := \max_{P \in \mathbb{P}} \mathrm{Precision}(L, P) \tag{7}$$

— which again can be computed with[15] (ipv^{with}) and without ($\mathrm{ipv}^{without}$) background knowledge. This comparison is done in Figure 5 by comparing the variance difference against the inverse purity difference

$$\mathrm{ipd}(L) := \frac{\mathrm{ipv}^{with}(L, \mathbb{P}) - \mathrm{ipv}^{without}(L, \mathbb{P})}{\mathrm{ipv}^{without}(L, \mathbb{P})}. \tag{8}$$

and against its linear interpolation. The diagram shows that the linear interpolation increases with decreasing variance difference.

We analyzed the categories that are not positively influenced by background knowledge. A detailed inspection shows that background knowledge does not improve variance for a few categories with extremely few members.[16] However, these categories do not affect (inverse) purity values because micro-averaging gives their purity values little weight. Furthermore, PRC has, among others, a category, i. e., the one labeled by ‘earn’, that could have been nicely classified in a supervised setting by signs that are semantically void (using, e.g., stop terms like ‘vs.’ which are not contained in Wordnet), that shows little variance anyway and that is therefore not improved by background representation. As ‘earn’ alone by far outnumbers all other categories in PRC (about 30% of all documents in PRC are categorized into ‘earn’), the inverse purity value increases only slightly, though significantly within the 0.5% confidence interval when background knowledge is added (see Table 3). In PRC-min15-max100 ‘earn’ is reduced to 100 members consequently decreasing its influence when micro-averaging.

4.5 Observations.

General observations of our experiments are that

[14] Observe that in Equation 5 both $\mathrm{var}(L)$ and $\mathrm{var}(D)$ change when background knowledge is incorporated.

[15] Using the same strategy with background knowledge mentioned before.

[16] 36 categories with less than 15 members are even left out of the figure.

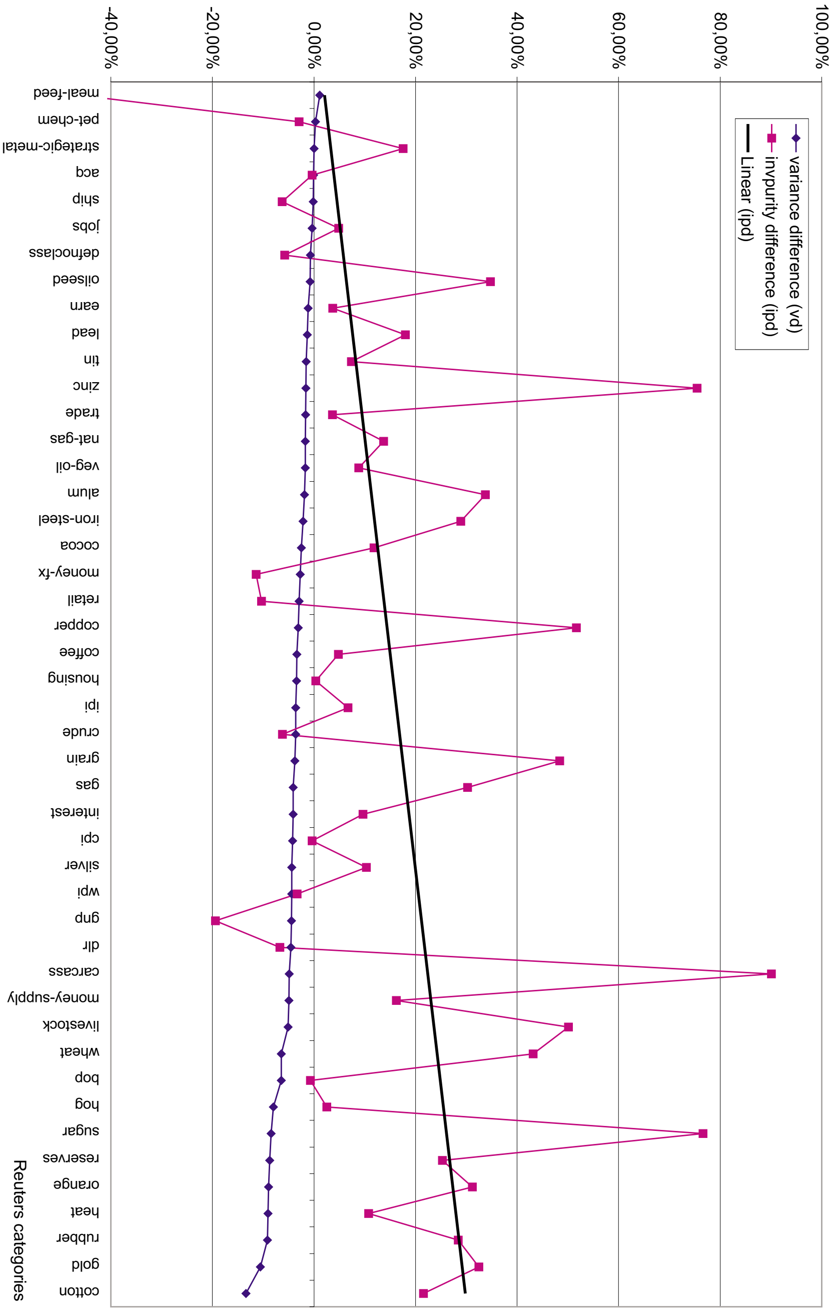

Fig. 5. Comparing the change of variance for each given category against the change of clustering results in terms of individual inverse purity values when the preprocessing strategy changes from best baseline to ‘standard’ (good) background knowledge (strategy hypdepth=5, hypint=add, hypdis=context, prune=30) on PRC with $k = 60$.

- background knowledge bought a lot of mileage if not too many documents had to be considered. The best results were achieved if categories had less than 20 documents. With an increase of documents per category the improvement by background knowledge decreased, however the good strategies were always as good as the baseline!
- background knowledge needs word sense disambiguation to be effective. However, extremely simple, generally applicable disambiguation strategies are already successful.
- Pruning with a threshold of 30 brought the best results. For categories with many documents pruning became less relevant.

Further evaluation values substantiating these results can be found at http://www.aifb.uni-karlsruhe.de/WBS/aho/clustering.

5 Conceptual Clustering of Texts and Text Clusters

Partitional clustering techniques as the ones discussed above have the disadvantage that they do not provide intensional descriptions of the clusters obtained. Conceptual Clustering techniques, on the other hand, provide such descriptions, but are known to be rather slow. In this section, we discuss a way of combining the advantages of both techniques.

Our approach consists of two steps. First, we apply Bi-Section-KMeans as described in the previous section in order to decrease the size of the problem. Each resulting cluster is considered as a ‘summary’ of similar documents, which will be treated as one object in the sequel. This step also incorporates the background knowledge as discussed above.

Then we cluster these objects using a conceptual clustering technique — in our case, Formal Concept Analysis. The latter provides intensional descriptions of the resulting clusters; and it is efficient enough, if the number of clusters chosen in the first clustering step is not too high. The resulting concept lattice can then be accessed using existing techniques from Formal Concept Analysis.

This section is composed of three parts. In Subsection 5.1, we recall the basic notions of Formal Concept Analysis. In Subsection 5.2, we explain how the document clusters are clustered conceptually. A discussion of exploration techniques and results is given in Section 5.3.

5.1 Conceptual Clustering by Formal Concept Analysis

As conceptual clustering technique, we make use of Formal Concept Analysis. Formal Concept Analysis (FCA) was introduced as a mathematical theory modeling the concept ‘concept’ in terms of lattice theory. This approach arose independently of ontologies, resulting in a different formalization of concepts. We discuss the differences below, after recalling the basics of Formal Concept Analysis as far as they are needed for this paper. An extensive overview is given in [8]. To allow a mathematical description of concepts as being composed of extensions and intensions, Formal Concept Analysis starts with a *formal context*:

Definition: A *formal context* is a triple $\mathbb{K} := (G, M, I)$, where G is a set of *objects*, M is a set of *attributes*, and I is a binary relation between G and M (i. e. $I \subseteq G \times M$). $(g, m) \in I$ is read “*object* g *has attribute* m”.

Let us first consider the following, simplified[17] situation: The set of objects consists of all documents, i. e., $G := D$, the set of attributes consists of all terms and concepts, i. e., $M := T \cup C$, and the relation I indicates if an attribute describes a document. How the relation is derived from the document representations will be discussed in Section 5.2 in detail. At the moment, consider only that we let $(d, t) \in I$ if document d 'is about' attribute t. For sake of simplicity, we will use 'attribute' as synonym for 'term or concept' and will denote it by t in the sequel; independent of the strategy chosen in Section 3.

Figure 6 shows a formal context with four texts as objects, and 80 terms and concepts as attributes, from which the first twelve are displayed.

	european	corpor	europ	central	american	market	washington	report	bank	drop	team	fire	...
Finance Text 1	×	×	×	×	×	×	×	×	×				
Finance Text 2						×	×	×	×				
Sports Text 1										×	×	×	×
Sports Text 2					×						×	×	×

Fig. 6. Formal context describing four texts.

From a formal context, a concept hierarchy, called *concept lattice*, can be derived:

Definition: For $A \subseteq G$, we define $A' := \{m \in M \mid \forall g \in A\colon (g, m) \in I\}$ and, for $B \subseteq M$, we define $B' := \{g \in G \mid \forall m \in B\colon (g, m) \in I\}$.

A *formal concept* of a formal context (G, M, I) is defined as a pair (A, B) with $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$. The sets A and B are called the *extent* and the *intent* of the formal concept (A, B). The *subconcept–superconcept relation* is formalized by

$$(A_1, B_1) \leq (A_2, B_2) :\Longleftrightarrow A_1 \subseteq A_2 \quad (\Longleftrightarrow B_1 \supseteq B_2) \ .$$

The set of all formal concepts of a context $\mathbb{K}$ together with the partial order $\leq$ is always a complete lattice,[18] called the *concept lattice* of $\mathbb{K}$ and denoted by $\underline{\mathfrak{B}}(\mathbb{K})$.

The extent of a formal concept consists thus of all objects belonging to the concept, and the intent consists of all their common attributes.

A possible confusion might arise from the double use of the word 'concept' in FCA and in ontologies. This comes from the fact that FCA and ontologies are two models for the concept of 'concept' which arose independently. In order to distinguish both notions, we will always refer to the FCA concepts as 'formal concepts' unless the meaning is clear from the context. The concepts in ontologies are referred to just as 'concepts' or as 'ontology concepts'. There is no direct counter-part of formal concepts in ontologies. Ontology concepts are best compared to FCA attributes, as both can be considered as unary predicates on the set of objects.

One of the twelve formal concepts which can be derived from the context given in Figure 6 is the tuple $\big(\{$ Finance Text 1, Finance Text 2 $\}$, $\{$ market, washington, report,

[17] In the sequel, we will use the clusters obtained by KMeans as objects, i. e., $G := \mathbb{P}$.

[18] I. e., for each set of formal concepts, there exists always a unique greatest common subconcept and a unique least common superconcept.

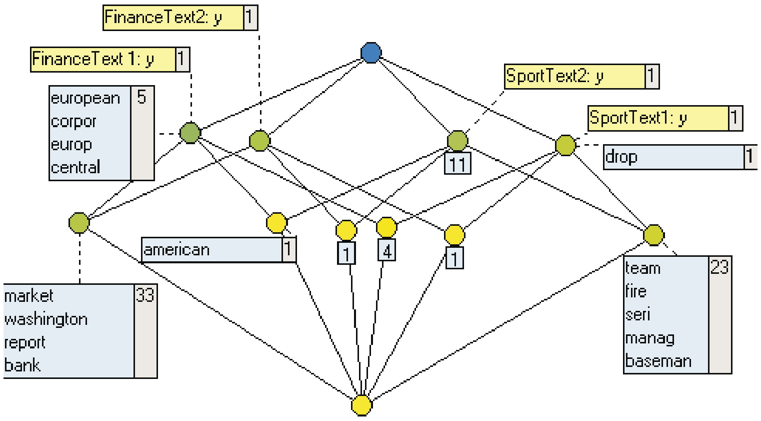

Fig. 7. Concept lattice of the context in Figure 6.

bank, [...] }), where [...] stands for 29 more attributes which are common to the two finance texts, but not to the two sports texts. These 29 attributes are not explicitly listed in Figure 6. Another formal concept — which is actually a subconcept of the first — is ({ Finance Text 1 }, { european, corpor, europ, central, american, market, washington, report, bank, [...] }) where [...] now stands for the 29 attributes from above plus four more attributes (which are not given explicitly in Figure 6 either) being shared by Finance Text 1 and Sports Text 1, but not by the two other texts.

Figure 7 shows a line diagram of the concept lattice derived from the formal context in Figure 6. The lattice was computed and visualized using the Cernato software of NaviCon Gmbh.[19] The first formal concept mentioned above is the left-most node in the diagram, the second concept the left-most node above it.

Line diagrams of concept lattices follow the conventions for the visualization of hierarchical concept systems as established in the international standard ISO 704. In a line diagram, each node represents a formal concept. Due to technical reasons we reverse, in this article, the usual reading order: A formal concept $\mathfrak{c}_1 \in \underline{\mathfrak{B}}(\mathbb{K})$ is a subconcept of a formal concept $\mathfrak{c}_2 \in \underline{\mathfrak{B}}(\mathbb{K})$ if and only if there is a path of ascending(!) edges from the node representing $\mathfrak{c}_2$ to the node representing $\mathfrak{c}_1$. The name of an object g is always attached to the node representing the most specific concept (i. e., the smallest concept with respect to $\leq$) with g in its extent (i. e., in our figure, the highest such node); dually, the name of an attribute m is always attached to the node representing the most general concept with m in its intent (i. e., the lowest such node in the diagram). We can always read the context relation from the diagram, since an object g has an attribute m if and only if the concept labeled by g is a subconcept of the one labeled by m. The extent of a concept consists of all objects whose labels are attached to subconcepts, and, dually, the intent consists of all attributes attached to superconcepts. The leftmost concept, for instance, has the two finance texts in its extent, and 33 attributes in its intent (from which the first four are listed).

In the diagram, the lowest node is always the concept having all objects in its extent and (in our case) no attributes in its intent. This is always the most general concept. This ‘all’-concept has six immediate subconcepts, each having exactly two objects in its extent,

[19] http://www.navicon.de

as there is no combination of words which describes a set of three objects, but not of the fourth one. Among these six concepts, we find the finance texts on the very left, and the sports texts on the very right. The second concept from the left is about the documents which are related to America. From the number of attributes attached to a concept, we can see the strength by which the documents are grouped together: the America concept depends on only one word ('american'), while the two sports texts are grouped together by 23 and the two finance texts are grouped together by 33 attributes, which indicates a much stronger relationship.

5.2 Clustering the Text Clusters with Formal Concept Analysis

As argued above, the computation of the concept lattice may be too time-consuming and/or the result too fine-grained for large sets of objects and/or attributes. Therefore, we now 'summarize' similar documents and treat them as one object in the sequel. 'Similar' means here being in the same Bi-Section-KMeans cluster.

Extracting Cluster Descriptions For applying a conceptual clustering approach like Formal Concept Analysis, we need intensional descriptions of the objects to be clustered. In our scenario this means that we have to determine the relation I, i. e., we have to decide, for each cluster and each attribute, if the attribute shall be considered as being important for the cluster or not. For performance reasons, we also would like to keep the total number of selected attributes small.

Therefore we need a method which points us to the most important attributes for each cluster. We followed an approach similar to the one described in [12]: We introduce a threshold θ to decide whether an attribute is important or not. This way we are also able to control how many attributes remain to describe the clusters. In our application, we used two thresholds, namely 15 % and 35 % of the maximal value. We could have used other, more sophisticated techniques for feature selection, as, e. g., described in [17,29]. But as feature selection is not our main research topic, we abstract from that aspect in this article.

We used the centroid vectors of the clusters for extracting the cluster descriptions. For each cluster, the description of the cluster is the set of all attributes having a value in the centroid vector which is above the threshold θ. This assures that those attributes are selected which are most important for the cluster. All attributes which were not assigned to at least one cluster were finally dropped. The assignment of the attributes to the clusters is the basis for the next step, the conceptual clustering part.

Being more precise, this approach looks as follows: The set of objects consists of all clusters determined in the previous step, i. e., $G := \mathbb{P}$. The set of attributes consists of all terms and concepts (which appear at least once), i. e., $M := T \cup C$. In the sequel, 'attribute' and 'term or cluster' are thus used synonymously. 'Object' is used synonymously with 'Bi-Section-KMeans cluster' unless otherwise stated. The relation I indicates if an attribute is related to a cluster, i. e., if its value in the centroid vector is above the threshold θ: $(P, t) \in I :\Longleftrightarrow (\vec{t_P})_t \geq \theta$.

In order to obtain a more fine-grained view, we additionally apply *conceptual scaling*. We may for instance use two (or more) thresholds in parallel. In the example in Subsection 5.3, we have for instance applied such an ordinal scale on the object set with two thresholds θ_1 and θ_2. The formal context (G, M, I) is then composed as follows: $G := \mathbb{P} \times \{\theta_1, \theta_2\}$, $M := T \cup C$, and $((P, \theta_i), t) \in I :\Longleftrightarrow (\vec{t_P})_t \geq \theta_i$. The relation I, applied to a pair (P, θ_i), returns thus the set $\{(P, \theta_i)\}'$ of all attributes which are more or less (i. e., with threshold θ_i) relevant for cluster P.

Once the formal context is set up, the concept lattice can be computed by one of the known algorithms (for an overview, see for instance [26]). We applied the Cernato tool mentioned above which makes use of B. Ganter's Next-Closure algorithm [7].

Partitional Clustering as Preprocessing for Conceptual Clustering Instead of considering the concept lattice as a clustering of clusters (i. e., as concept lattice of the context $(\mathbb{P}, T, I)$), the resulting concept lattice can also be interpreted as a concept hierarchy directly on the documents, as it is isomorphic to the concept lattice of the context (D, T, J) with $(d, t) \in J$ iff $d \in P$ and $(\vec{t_P})_t \geq \theta$ for some cluster $P \in \mathbb{P}$. This context is in fact an approximation of the descriptions of the documents by term vectors [20] with the property that all documents in one cluster obtain exactly the same description. This loss of information is the price we pay for improving the efficiency. [21]

An advantage of using the partitional clusters as intermediate step, however, is that we can deal with new, previously unseen documents in a robust way: A new document is first assigned to the cluster with the closest centroid, and then finds its place within the concept lattice. If on the contrary the document would be considered directly for computing the concept lattice, it could not be guaranteed that the structure of the lattice would not change when a new document arises.

5.3 Exploring the Conceptual Structure of the Document Collection

Line diagrams of concept lattices provide a visualization of the clustering obtained. An interactive exploration of the diagrams supports the user in analyzing the conceptual structure inherently present in the document collection.

In this section, we present two examples which show up the potential of the approach. Both are based on the dataset PRC-min15-max100 described in Section 4.3. Partitional clustering was performed with Bi-Section-KMeans with the following parameters and strategies: number of clusters $k = 60$, Ontology = true, Hypdepth = 5, Hypdis = first, Hypint = only, and weighting with tfidf using pruning threshold 30. We obtained a clustering with 60 clusters, having a purity of 59,1 % (which is close to the average of the 20 runs we performed). Based on this clustering, we extracted the important ontology concepts for each cluster descriptions (with thresholds $\theta_2 = 15\,\%$ and $\theta_2 = 35\,\%$) and computed the concept lattice.

Exploring the Concept Lattice The concept lattice is shown in Figure 8. The clusters are named CL 0 to CL 59, and the number in brackets behind each name indicates how many documents the cluster contains. 'm' stands for 'medium', i. e., for $\theta_1 = 15\,\%$ of the maximal value, and 'h' stands for 'high', i. e., for $\theta_2 = 35\,\%$ of the maximal value. For instance, 'CL 19 - (67): m' represents Cluster 19 which consists of 67 documents; and all ontology concepts which have at least medium importance for this document (e. g., 'food, nutrient') can be found below in the diagram.

[20] I. e., of the formal context $(D, T, \tilde{J})$ with $(d, t) \in \tilde{J}$ iff $d \in P$ and $(\vec{t_d})_t \geq \theta$. This context however would result, after a very time-consuming computation, in a far too large concept lattice.

[21] In the worst case, the complexity of the resulting concept lattice is exponential in the size of G (i. e., of D or $\mathbb{P}$, whereby $|\mathbb{P}| \ll |D|$ in practice).

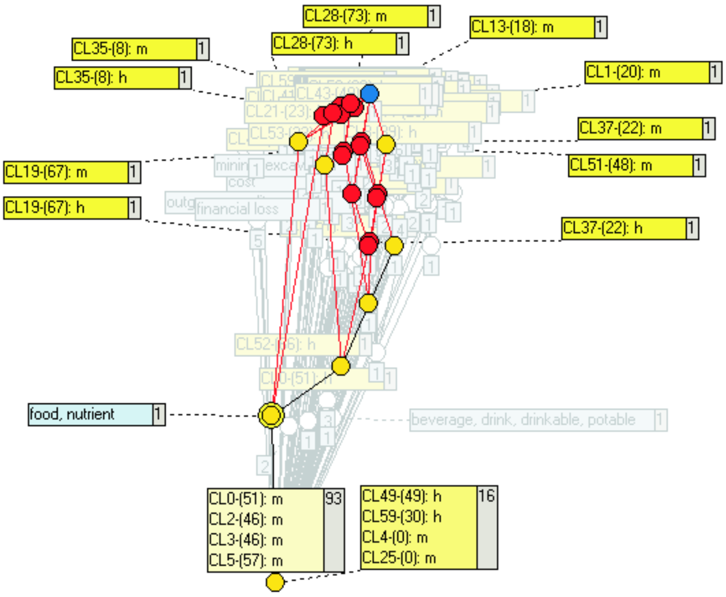

Fig. 8. The total concept lattice (highlighting the formal concepts related to food).

Even though we have performed partitional clustering before, the concept lattice is usually too large to be completely displayed at once. A typical approach for exploring the lattice is to study the top-level concepts (i. e., those close to the bottom of the diagram) first. In the diagram, we have highlighted the top-level concept food, together with all its subconcepts. We can see that food has a high (and hence also medium) occurrence in clusters 19, 28, 35, and 37; and medium occurrence in clusters 1, 13, and 51. In order to analyze these clusters in more detail, we can restrict the set of objects to these clusters, and visualize only that part of the concept lattice. The resulting sub–semilattice is shown in Figure 9.

Starting with the top formal concepts again (i. e., with the bottom part of the figure), we discover that our clusters belong to three overlapping topics: beverages, food and sugar/food ingredients. For the latter, our knowledge representation is not detailed enough to decide whether all these documents are about sugar as a food ingredient, or if the partitioning algorithm grouped together documents about sugar with documents about food ingredients. If one is interested in this set of clusters in more detail, one must have a closer look either by analyzing the centroid vectors, or by directly browsing the documents.

As for the beverages, one discovers that there are two main beverages mentioned, coffee and cocoa. Among the clusters dealing with coffee, there are some about South American countries. Among them, Brazil and Colombia are most relevant, but there are more countries of that region, which are for instance mentioned in documents of Cluster 19. More details can be read from the diagram.

We want to emphasize that the resulting conceptual clustering benefits heavily from the use of background knowledge: The attributes ‘food’, ‘beverage’, etc. are not contained explicitly in the documents, and hence neither in the cluster descriptions unless we make use of the background knowledge (i. e., using one of the strategies ‘add’, ‘repl’, or ‘only’

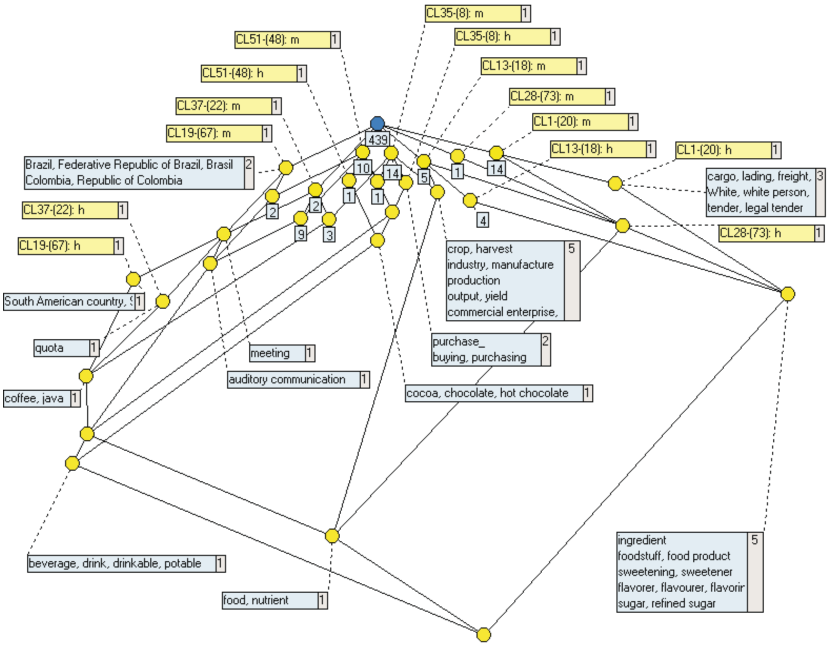

Fig. 9. The concept lattice restricted to the formal concepts related to food.

from Section 3.2). Without this additional input, the concept lattice would be completely flat, and would not support the discovery of conceptual relationships between the clusters determined by the partitional clustering algorithm.

Using the Similarity Measure to Find Interesting Parts Another way of starting the exploration of the conceptual structure of the document collection is to focus on some clusters which are indicated to be similar according to the similarity measure used in the partitional clustering step as described in Section 4.

In order to give a first hint where to discover interesting structures, we applied first a magnetic spring algorithm for graph visualization[22] for recognizing which clusters are related. A part of the resulting graph is shown in Figure 10. Based on the cosine similarity, it tries to map the clusters into the Euclidean plane such that clusters with similar centroids attract each other, and clusters with different centroids repel each other. Strong similarity (with respect to a given threshold, in our example 25 % of the maximal similarity) is indicated by a line between the clusters. The term in parentheses behind a cluster name in the diagram indicates to which Reuters topic the majority of the documents in the cluster were assigned by the Reuters experts. Of course one does not have this additional information when clustering documents in an unsupervised way. We added this information for simplifying the evaluation. In an unsupervised setting, one could display the most important attribute(s) describing the cluster instead.

In the diagram, we see for instance that the Clusters 39, 12, 16, 30, 53, 38, 10, and 34 in the upper right part of the figure have similar centroids. In order to analyze the similarity

[22] http://java.sun.com/applets/jdk/1.0/demo/GraphLayout/

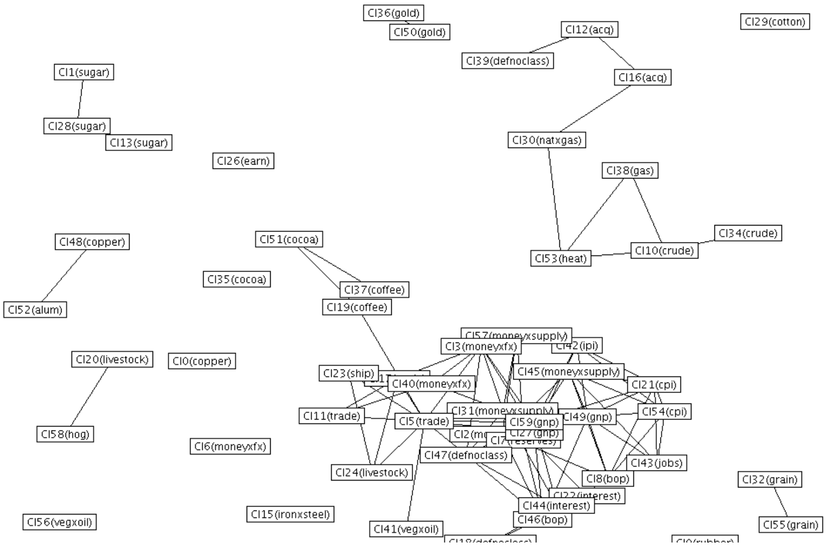

Fig. 10. Graph showing (distance-based) similarities between the text clusters

of this group of clusters conceptually, we restrict the object set of the formal context to just those clusters, and re-compute the concept lattice. In order to discover the general structure before going into too much detail, we selected only one threshold, $\theta = 15\,\%$ of the maximal value (indicated by 'y' in the diagram). The resulting concept lattice is again a sub–semilattice of the concept lattice in Figure 8. It is shown in Figure 11.

Starting from the top level (i. e., the bottom part of the figure) again, we see that there are three main areas: crude oil, gas, and business. The first two of them overlap in Cluster 53, which groups together documents where the use of oil or gas for heating is addressed. In fact, the majority of documents in this cluster was assigned to the 'heat' label by the Reuters experts (see Fig. 10). From the concept lattice we can read that Clusters 10, 34 and 38 are about crude oil (and Clusters 10 and 34 additionally about (its) transport); which coincides for the first two clusters with the Reuters classification to 'crude' for the majority of documents in these clusters. Most of the texts in Cluster 38 contain the word 'gasoline', which justifies its classification under the more general attribute 'oil'. The concept lattice shows that Cluster 30 is about gas as a natural phenomenon, which coincides with the assignment of most of its texts to the label 'natxgas' by the Reuters experts.

The third top level concept, labeled by 'business', is disjoint to the two other top level concepts. And in fact, the majority of the documents in clusters 12 and 16 were assigned by the Reuters experts to a different label, namely 'acq[uisition]'. When checking the concept intent of the concept labeled by 'CL 39', one observes a large diversity of topics, e. g., issue, vehicle, park, document, security, share, business. When reading the documents in this cluster, one observes that they cover indeed many unrelated topics. The fact that our approach could not provide a clear description of this cluster correlates with the fact that the Reuters experts could not do better, as they could not assign meaningful labels to the majority of documents in Cluster 39 either — most of them have the label'defnoclass'.

This last example shows that one can also identify inconsistencies in the results of partitional clustering by using Formal Concept Analysis. We observed during our exploration

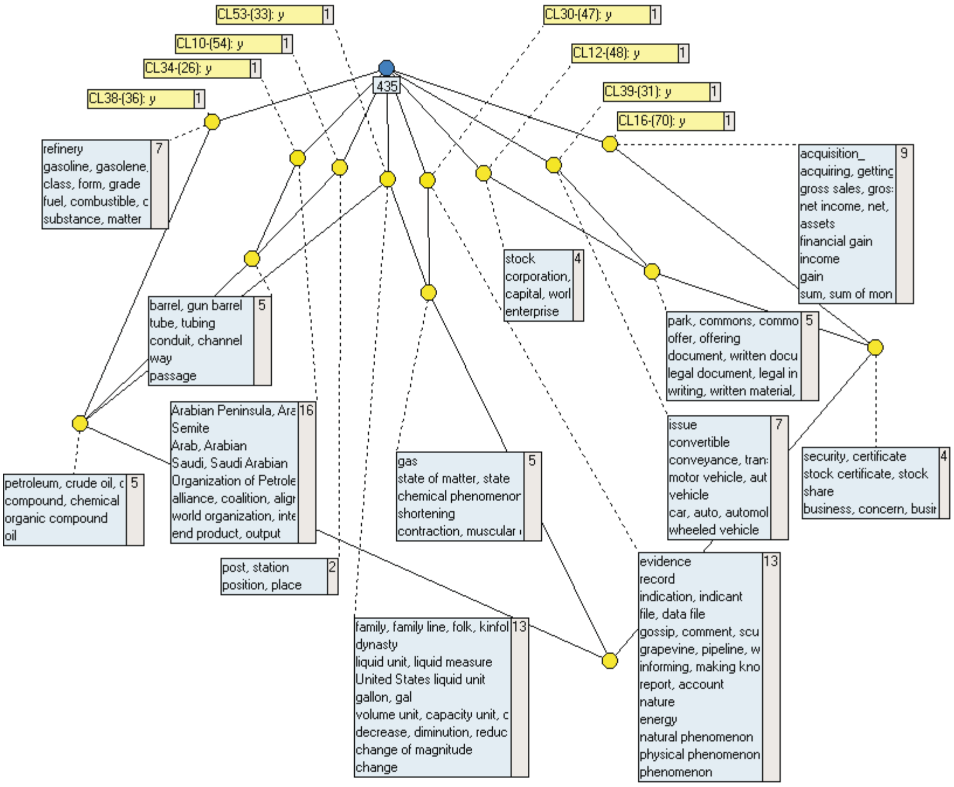

Fig. 11. Concept lattice focusing on the clusters 10, 12, 16, 30, 34, 38, 39, and 53.

of the concept lattice that humans deal quite well with noise (i. e., words which do not help to understand the clustering) which may have been accidentally in the preprocessing steps, due to the presentation of the results on a semantical level by the concept lattice.

6 Discussion

Let us discuss the major insights that we have gained from our experiments:

Background knowledge helps. The best strategies (e.g., hypint = add, hypdis = context, hypdepth = 5) can be safely used, as they are frequently improving and never degrading performance compared to the best baseline.

The principal idea of our approach is that the variance of documents within one category is reduced by representation with background knowledge, thus improving results of text clustering measured in terms of purity and inverse purity with conventional means like Bi-Section-KMeans. To this end, different, but semantically similar terms in two text documents may contribute to a good similarity rating if they are related via Wordnet synsets or hypernyms. The vector representations are thereby re-adjusted such that the angles in-between term vectors containing these hypernyms decrease, i.e. their similarities increase.

Our results indicate that this effect is particularly relevant, if terms occur infrequently in the corpus (as is typically the case when categories have few documents only).

In the Reuters corpus we have seen that variance reduction is achieved and effective for most categories, with one exception, the ‘earn’ category (cf. Section 13), which may best be clustered by syntactic indicators like ‘vs.’, rather than semantic representations. Our assumption is that in other real-world applications, one will also find categories that should rather be clustered by syntactic means as well as categories that would rather profit from background knowledge. As we did not significantly decrease performance on the former ones, but clearly improved performance on the latter, the conclusion from PRC and its derivations is that one should always use background knowledge for text clustering (and eventually also consider further syntactic means).

Background knowledge strongly benefits from feature weighting. Adding background knowledge may not be beneficial for the clustering task *per se*. Rather, hypernyms at the higher levels may even detriment the clustering performance (as shown by $r = 5$ without tfidf weighting in Figure 3).

However, such hypernym representations constitute “systematic noise”. Therefore, if a hypernym occurs often *and* is too general to be helpful, then its effect on the representation may be outweighed by feature weighting. To some extent this may even be the case should the hypernym be misleading.

Only if a hypernym occurs with low or medium frequency and it is misleading considering the original term it represents, then it appears to be detrimental. The reader may note that a resource like Wordnet has been carefully built in order to reflect common understanding. Errors in the basic level concepts, i.e. the ones in the middle that are neither very general nor very specific, or misleading re-interpretations of the concepts by the writer of a text do not seem to occur too often.

Word sense disambiguation helps. While Wordnet contributes additional knowledge, important distinctions may be blurred without word sense disambiguation. There seems to be the tendency that vector re-adjustments are too strong without a selection of a particular concept.

We could show that some kind of word sense disambiguation is needed in order to avoid the incorporation of too much noise. We conjecture that the inclusion of more refined strategies such as known from the literature will further improve the clustering results.

Conceptual clustering adds explanatory power to the cluster results. The major advantage of conceptual clustering is the intensional description coming along with each cluster. While some intensional description can also be obtained by post-processing the partional clustering results (e. g., by just applying our attribute assignment described in Section 5.2), conceptual clustering provides additional benefits. First, it derives more general clusters and arranges them in a hierarchy which is consistent with the cluster descriptions (and with the hierarchical organization of the background knowledge). Second, it allows for multiple inheritance (unlike many hierarchical clustering techniques) which reflects the human way to structure conceptual knowledge. And third, being a lattice, it allows to compute, for each set of clusters, the unique least common supercluster and the greatest common subcluster. This allows the computation of dependencies between important terms in the document collection, and algorithmic support of navigation and retrieval tasks.

Navigation by Formal Concept Analysis is suitable to support the novice data mining expert. Even though we did not perform systematic user studies, our experiments (performed by an FCA novice) showed that the visualization by concept lattices and the interaction with them offer an intuitive access to the clustering results. Especially the top-down access from quite general to more specific (formal) concepts was considered as very natural and straightforward.

The combination of partitional and conceptual clustering works. While producing results as good as or better as the baseline, our experiments show that the explanatory power (which is hard to measure by 'objective' measures) is improved. As from a performance point of view, our observations show that we are indeed able to derive clusters of objects together with intensional descriptions in reasonable time; and still with a reasonable degree of detail. In particular, the background knowledge added in the first place to improve partitional clustering is also fully exploited by conceptual clustering. For instance, this can be seen by major intensional descriptions like 'food' that are derived and highly ranked without appearing in the texts at all.

Further improvements. Further improvements by background knowledge might still be achieved by exploiting syntactic taggers and all of Wordnet including adjectives and verbs. For instance, the term "international" in "an international company" has not been recognized to be an adjective by our approach. Instead it has been used assuming that it stands for a noun as in "the communist international".

The purity and inverse purity measures we have applied for evaluation only consider the first Reuters label. Our experiments however showed that even when the clustering did not map to this first label, it mostly got the second (and sometimes third) label correct. Since a standard measure that properly accounts for multi-labeling does not exist, there is a need to modify the (inverse) purity measure accordingly.

Thus, we have only reaped the — comparatively — low hanging fruits, improving standard clustering for many relevant tasks while leaving many possibilities (e.g., weighting, disambiguation) for further probable improvements.

7 Related Work

While we do not know of any research that exploits background knowledge for text document clustering, there are a number of related uses.

Wordnet has mostly been used in information retrieval and in supervised learning scenarios up to now (but to our knowledge not for clustering): In information retrieval, Voorhees [28] as well as Moldovan and Mihalcea [18] have explored the possibility to use Wordnet for retrieving documents by keyword search. It has already become clear by their work that particular care must be taken in order to improve precision and recall.

Buenaga Rodríguez et. al. [6] and Ureña Lóez et. al. [27] show a successful integration of the Wordnet resource for a document categorization task. They use the Reuters corpus for evaluation and improve the classification results of the Rocchio and Widrow-Hoff algorithms by 20 points. In [9], Wordnet is used for word sense disambiguation. Gonzalo et. al. show in an information retrieval setting the improvement of the disambiguated synset model over the term vector model. In contrast to our approach, [6], [27], and [9] apply Wordnet to a supervised scenario (and not to an unsupervised one as in our application), do not make use of Wordnet relations such as hypernyms, and build the term vectors manually.

Approaches like term clustering [12], LSI [5] or PLSI [10] use statistic methods to compute a kind of “concepts”. These concepts are rather different to our definitions of ontology concepts and formal concepts. They are not able to indicate the meaning of the concepts and there exists no understandable mapping to lexical entries. A generalization of their “concepts” is not possible. We do not know of actual comparisons that relate KMeans or Bi-Section-KMeans with LSI or PLSI using the same dataset for clustering.

We have built our numerical comparisons on Bi-Section-KMeans which has proved to be very robust in a wide variety of experiments [24]. Also to our experience it performed as good as other algorithms that we tested informally. Its standard parameter settings evaluated as good as other ones (e. g., bi–secting based on variance instead of cardinality; cf. [24]).

Our extraction of descriptions for the text clusters has been inspired by [12]. There, Karypis and Han show that cluster centroids can be used to summarize the content of a cluster. They state that the most important terms in a cluster centroid are the terms with the highest weight. This observation underlies our approach in Section 5.2, where we use only the highly weighted terms to describe the content of the cluster. They differ from our approach in that they select the ten best terms while we consider all terms above a given threshold. We additionally make use of Wordnet.

Conceptual clustering with Formal Concept Analysis has been discussed in [25,4,16,26]. Another approach to Conceptual Clustering is for instance discussed in [14]. Formal Concept Analysis differs from them in that it does not make use of any heuristics (including arbitrary start settings) and allows for overlapping clusters. Compared to non-conceptual clustering approaches, all conceptual clustering approaches have in common less computational efficiency. Our paper is an approach to overcome this drawback.

8 Conclusion

In this paper, we have discussed a way of combining the efficiency of a partitional clustering technique, the expressivity of background knowledge, and the explanatory power provided by a conceptual clustering approach. We first changed the text document representation to accomodate background knowledge, then we clustered the documents using Bi-Section-KMeans, and eventually we performed conceptual clustering with Formal Concept Analysis.

Our empirical evaluation showed the benefit of using background knowledge. It also showed that the combination of efficient partitional clustering with expressive conceptual clustering techniques allows to exploit the advantages of both approaches.

Acknowledgements. We thank our colleagues Alexander Maedche and Viktor Pekar for many fruitful discussions during the work on this paper. This work has been supported by EU in the IST project Bizon and by BMBF in the project PADLR.

References

1. E. Agirre and G. Rigau. Word sense disambiguation using conceptual density. In *Proc. of 15th International Conference on Computational Linguistics, COLING’96. Copenhagen, Denmark, 1996*, 1996.

2. G. Amati, C. Carpineto, and G. Romano. Fub at trec-10 web track: A probabilistic framework for topic relevance term weighting. In *The Tenth Text Retrieval Conference (TREC 2001)*. National Institute of Standards and Technology (NIST), online publication, 2001.
3. E. Bozsak, M. Ehrig, S. Handschuh, A. Hotho, A. Maedche, B. Motik, D. Oberle, C. Schmitz, S. Staab, L. Stojanovic, N. Stojanovic, R. Studer, G. Stumme, Y. Sure, J. Tane, R. Volz, and V. Zacharias. Kaon - towards a large scale semantic web. In *Proceedings of EC-Web*, pages 304–313, Aix-en-Provence, France, 2002. LNCS 2455 Springer.
4. C. Carpineto and G. Romano. GALOIS: An order-theoretic approach to conceptual clustering. In *Machine Learning, Proc. ICML 1993*, pages 33–40. Morgan Kaufmann Publishers, 1993.
5. S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.
6. M. de Buenaga Rodrıguez, J. M. Gomez Hidalgo, and B. Díaz-Agudo. Using WordNet to complement training information in text categorization. In *Recent Advances in Natural Language Processing II*, volume 189. John Benjamins, 2000.
7. Bernhard Ganter. Algorithmen zur formalen begriffsanalyse. In B. Ganter, R. Wille, and K. E. Wolff, editors, *Beiträge zur Begriffsanalyse*, pages 241–254. B.I.–Wissenschaftsverlag, Mannheim, 1987.
8. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin – Heidelberg, 1999.
9. J. Gonzalo, F. Verdejo, I. Chugur, and J. Cigarrán. Indexing with WordNet synsets can improve text retrieval. In *Proceedings ACL/COLING Workshop on Usage of WordNet for Natural Language Processing*, 1998.
10. T. Hofmann. Probabilistic latent semantic indexing. In *Research and Development in Information Retrieval*, pages 50–57, 1999.
11. N. Ide and J. Véronis. Introduction to the special issue on word sense disambiguation: The state of the art. *Computational Linguistics*, 24(1):1–40, 1998.
12. G. Karypis and E. Han. Fast supervised dimensionality reduction algorithm with applications to document categorization and retrieval. In *Proc. of the 9th ACM International Conference on Information and Knowledge Management CIKM-00*, pages 12–19. ACM Press, New York, US, 2000.
13. D.D. Lewis. Reuters-21578 text categorization test collection, 1997.
14. R. S. Michalski and R. Stepp. Learning from observation: Conceptual clustering. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning, An Artificial Intelligence Approach*, volume II, pages 331–363, Palo Alto, 1983. TIOGA Publishing Co.
15. G. Miller. WordNet: A lexical database for english. *CACM*, 38(11):39–41, 1995.
16. G. Mineau and R. Godin. Automatic structuring of knowledge bases by conceptual clustering. *IEEE Transactions on Knowledge and Data Engineering*, 7(5):824–829, 1995.
17. D. Mladenic and M. Grobelnik. Feature selection for classification based on text hierarchy. In *Working Notes of Learning from Text and the Web, Conf. Automated Learning and Discovery (CONALD-98)*. Carnegie Mellon Univ., Pittsburgh,, 1998.
18. D. I. Moldovan and R. Mihalcea. Using WordNet and lexical operators to improve internet searches. *IEEE Internet Computing*, 4(1):34–43, 2000.
19. Patrick Pantel and Dekang Lin. Document clustering with committees. In *Proc. of SIGIR'02, Tampere, Finland*, 2002.
20. M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.
21. G. Salton, A. Wong, and C.S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18:613–620, 1975.
22. G. Salton. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley, 1989.
23. Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

24. M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 2000.
25. S. Strahringer and R. Wille. Conceptual clustering via convex-ordinal structures. In O. Opitz, B. Lausen, and R. Klar, editors, *Information and Classification*, pages 85–98, Berlin-Heidelberg, 1993. Springer.
26. G. Stumme, R. Taouil, Y. Bastide, N. Pasqier, and L. Lakhal. Computing iceberg concept lattices with titanic. *J. on Knowledge and Data Engineering*, 42(2):189–222, 2002.
27. L. A. Ureña Lóez, M. de Buenaga Rodríguez, and J. M. Gómez Hidalgo. Integrating linguistic resources in tc through wsd. *Computers and the Humanities*, 35(2):215–230, 2001.
28. E. M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of ACM-SIGIR. Dublin, Ireland*, pages 61–69. ACM/Springer, 1994.
29. Y. Yang and J. O. Pederson. Feature selection in statistical learning of text categorization. In *Proc. of the 14th International Conference of Maschine Learning ICML97*, 1997.