Explaining Text Clustering Results using Semantic Structures

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

Abstract. Common text clustering techniques offer rather poor capabilities for explaining to their users why a particular result has been achieved. They have the disadvantage that they do not relate semantically nearby terms and that they cannot explain how resulting clusters are related to each other. In this paper, we discuss a way of integrating a large thesaurus and the computation of lattices of resulting clusters into common text clustering in order to overcome these two problems. As its major result, our approach achieves an explanation using an appropriate level of granularity at the concept level as well as an appropriate size and complexity of the explaining lattice of resulting clusters.

1 Introduction

Clustering is an important task that is performed as part of many text mining and information retrieval systems. Clustering can be used for efficiently finding the nearest neighbors of a document [1], for improving the precision or recall in information retrieval systems [15, 11], for aid in browsing a collection of documents [3, 8], for the organization of search engine results [19], or for the personalization of search engine results [13].

Most current document clustering approaches are based on the vector-space model (also called bag of words model or word space), the dimensions of the vector space are constituted by the important words of the document collection. The respective term or word frequencies (TF) in a given document constitute the vector describing this document. In order to discount frequent words with little discriminating power, each word can additionally be weighted based on its Inverse Document Frequency (IDF) in the document collection. Once the documents are mapped into the vector space, they can be clustered according to the distances between the vectors. However, what is neglected in these approaches are the explanations of why particular clusters have been formed and how the different clusters are related to each other.

To elaborate on this, we build on a finding by Karypis and Han that we could confirm. In [10], they have shown that words occurring with high weights in the centroid of a cluster can be used to summarize the content of the cluster. Thereby, they observe that "prevalent terms of the various centroids often contain terms that act as synonyms within the context of the topic they describe." Common text clustering algorithms now lack the capabilities

(A) to recognize such synonyms in order to improve text clustering quality;

(B1) to use such synonymity in order to improve the quality of the explanation of why a cluster has been formed (e.g., just state 'this cluster is about Volkswagen' instead of stating 'this cluster is about Volkswagen and about VW');

(B2) to exploit semantic hierarchies of words in order to abstract an explanation (e.g., instead of 'this cluster is about pork and beef and veal' state 'this cluster is about meat');

(C) to give an account of how resulting clusters are related (e.g. 'cluster1 is about the same topics as cluster2, but additionally about meat').

With regard to *(A)*, we have investigated how a large thesaurus like WordNet [5] may help to improve text clustering results exploiting synonymity and other semantic relationships.[1]

With regard to *(B1)* and *(B2)*, we make use of the synonymity of words and the hierarchy of their corresponding concepts as defined in a thesaurus[2] in order to come up with more concise and abstracting explanations. We extract the explanations from the centroid representation of resulting clusters, but we also use thesaurus concepts instead of words only.

Finally, with regard to *(C)*, we have explored the use of lattice theory. Formal concept analysis [6] computes the place of an object representation (e.g. the representation of a text or the representation of a text cluster) in a lattice according to its vector representation. Unfortunately, formal concept analysis (and similar means of analysis) are not suited to directly relate vector representations of large collections of texts. Tests with text samples have revealed that even homogeneous text sets have relatively few joint word occurrences leading to large and complex lattices with unsatisfying explanatory power.[3]

In this paper, we present an approach addressing the challenges listed above. Here, we will specifically discuss the challenges *(B)* and *(C)*. Our approach proceeds along the following lines:

1. It represents text documents by a vector model that exploits the hierarchy of the concepts in the WordNet thesaurus (cf. Section 2);
2. it uses a common text clustering algorithm, BiSec–k–Means, to aggregate texts without supervision into a pre-defined number of clusters. Moreover it extracts a representation of each resulting cluster (cf. Section 3);
3. it computes a lattice from the resulting cluster representations to relate, *(i)* words from the thesaurus hierarchy with the different clusters and, *(ii)* to compare the different cluster representations (cf. Section 4);
4. eventually, it visualizes (parts of) the resulting lattice structure(s) to the user allowing exploration and explanation of how and why clustering results have been produced (cf. Section 5).

[1] A comprehensive empirical investigation on how a thesaurus may improve text clustering is reported in [9].

[2] What is typically called a 'concept' in ontologies or thesauri is a very close match to what is called a 'synset' in WordNet.

[3] In addition, result size complexity of formal concept analysis may become a problem if the lattice is computed on several thousands of texts, as the number of nodes in the lattice may grow exponentially with the number of objects. In practice runs, however, we have observed that formal concept analysis scaled quite well with regard to runtime and the size of the resulting lattice.

We discuss our approach along the Reuters–21578 text collection. Section 6 provides an overview of related work. In particular, we emphasize that while the individual algorithms of steps 1 through 4 are well-known and to some extent interchangable with likewise approaches, the combination we describe here is unique and original, while it serves frequently arising objectives in text clustering.

2 Representing Texts

This section describes the representation of texts in an exemplary manner drawing from the Reuters-21578 dataset on which we performed many of our experiments. The basic idea of this section is the extension of the common text representation as a vector in a word space towards a representation in a word/concept space.

The Reuters-21578 Dataset We selected the Reuters-21578[4] text collection for our experiments. The corpus consists of 21578 documents. This corpus is especially interesting for evaluation, as part of it comes along with a (hand-crafted) classification. It contains 135 so-called topics. To be more general, we will refer to them as 'classes' in the sequel. For allowing evaluation, we restrict ourselves to the 12344 documents which have been classified manually by Reuters. Some of them could not be assigned by the experts to one of the predefined classes; we collect them in an additional class 'defnoclass'. Reuters assigns some of its documents to multiple classes, but we consider only the first assignment. After these steps, we obtain our final corpus $\mathcal{D}$ for evaluation. It consists of the 12344 documents, distributed over 82 Reuters topics.

Preprocessing the Document Set For the preprocessing of the documents, we used the text mining system developed at AIFB within the KAON[5] framework. We performed the following steps on the selected corpus: First we lowered the letters of all words and removed stopwords. We used a stopword list with 571 entries which removed 416 stopwords from the documents. We also dropped all words with less than 30 occurrences over the whole corpus. 17917 words were removed in total. After these steps, 2657 different words remained in our list, with a total occurrence of 784434.

WordNet as Background Knowledge Instead of using a bag-of-word model directly, we additionally enriched the text representation with background knowledge. The basic idea is to replace the words by concepts and their broader concepts as defined in a given thesaurus, in order to capture similarities at various levels of generalization. For this purpose we needed a resource suitable for the Reuters corpus. We choose WordNet[6] as our background knowledge. WordNet consists of so-called synsets, together with a hypernym/hyponym hierarchy.[7]

To modify the existing word vector representations of text, we have first replaced all nouns that appeared in the documents and that were known by WordNet by the corresponding concept ('synset') identifiers from WordNet. At this point, we had several

[4] http://www.daviddlewis.com/resources/testcollections/ reuters21578/
[5] http://kaon.semanticweb.org
[6] http://www.cogsci.princeton.edu/~wn/
[7] See http://www.cogsci.princeton.edu/~wn/man1.7.1/wngloss.7WN.html for a glossary.

choices of, *(i)*, how to deal with terms not known by WordNet (delete or keep), *(ii)*, how to deal with ambiguity (one word in the document, like 'bank', may correspond to several concepts in WordNet), and, *(iii)*, how many generalizations of a concept to consider to use for the text representation. We have elaborated on these choices in [9] and present here only a simple, but quite effective combination.

In the simplest case, *(i)*, we have ignored all words that either were not nouns or that were not known by WordNet. *(ii)*, we have used a disambiguation method provided by WordNet. WordNet has a ranking of what is the 'most common' meaning for a word in English. We here use only this static ranking to map a word onto corresponding concepts. *(iii)*, we have mapped a word occurrence in a document to its most highly ranked concept in WordNet as well as to the four most specific generalizations of this concept. For instance, the occurrence of 'bank' in a document would increase the vector entry corresponding to 'banking company' (the concept) as well as the vector entries corresponding to 'financial institution', 'institution', 'organization' and 'social group' as these are the four most specific generalizations of 'banking company'. The concepts that were assigned to at least one document formed then the new set $\mathcal{T}$ of terms used for describing the documents, i. e., they constitute the dimensions of the vector space for the new text representation.

Enriching the term vectors with concepts from WordNet has two benefits. First it resolves synonyms; and second it introduces more general concepts which help identifying related topics. For instance, a document about 'bank' may not be related to a document about 'insurer' by the cluster algorithm if there are only 'bank' and 'insurer' in the term vector. But if the more general concept 'financial institution' is added to both documents, their semantical relationship is revealed.

In the remainder of this paper, we will use the expression 'term' both for words and for concepts (synsets) of the thesaurus for sake of simplicity. If we talk about one of them specifically, we will mention it explicitly.

Building the Term Vectors Based on the work done so far, we built a term vector for each document $d \in \mathcal{D}$. For each document, the terms $t \in \mathcal{T}$ are weighted by *tfidf* (term frequency $\times$ inverse document frequency) [16], which is defined as follows: $\textit{tfidf}(d,t) = \textit{tf}(d,t) \times \log\left(\frac{|\mathcal{D}|}{|\mathcal{D}_t|}\right)$, where $\textit{tf}(d,t)$ is the frequency of term t in document d, and $\mathcal{D}_t \subseteq \mathcal{D}$ is the set of all documents containing term t. The term vector for document d is then the tuple $\boldsymbol{w}_d := (\textit{tfidf}(d,t))_{t\in\mathcal{T}}$.

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 hold the balance and, hence, are expected to be better able to contribute to clustering results.

After this first step, we have thus obtained a description of all documents, which is enriched by background knowledge, and which will also allow to relate semantically close (but syntactically different) documents.

3 Text Clustering and Feature Extraction

In this section, we show how to cluster our, so far uncommon, text representations with state-of-the-art methods (cf. [17]). The major output of this section is an explanation of

the text clusters achieved by known means, like [10], which is then used as an input for analysis and explanations in subsequent sections.

The reader may note that while we present a specific, effective and efficient, approach for text clustering and extraction of cluster representations here, this approach might be replaced by other methods that digest similar input and produce similar output without changing our principal approach.

3.1 Text Clustering with BiSec–k–Means

On the preprocessed data we applied BiSec–k–Means [17], a 'bisecting' variant of k–Means, using the cosine similarity: the similarity between two documents $d_1, d_2 \in \mathcal{D}$ is calculated as the cosine of the angle between their term vectors $\boldsymbol{w}_{d_1}$ and $\boldsymbol{w}_{d_2}$.

For this clustering step we need a fast algorithm (such as k–Means), which is able to deal with large datasets, which should also provide a reasonable accuracy. Instead of a slow agglomerative clustering technique with a good accuracy we choose BiSec–k–Means which tends to give better results as k–Means and is sometimes also better as agglomerative clustering, while it is as fast as k–Means (cf. the seminal paper [17]).

BiSec–k–Means is based on the k–Means algorithm. It repeatedly splits the largest cluster (using k–Means) until the desired number of clusters is obtained. As input, it takes the list $(\boldsymbol{w}_d)_{d\in\mathcal{D}}$ of document descriptions, and the number k of desired clusters. As output, it provides a partitioning $\mathcal{C}$ of of the set $\mathcal{D}$ of documents (i. e., a set $\mathcal{C}$ of k disjoint subsets of $\mathcal{D}$ with $\bigcup_{C\in\mathcal{C}} C = \mathcal{D}$). Each cluster $C \in \mathcal{C}$ is represented by its centroid $\boldsymbol{w}_{\boldsymbol{C}}$.

3.2 Extracting Cluster Descriptions

For a good explanation of results, it is necessary to detect important terms and be concise about the explanation created. The basic idea of mechanisms like latent semantic indexing (LSI) [4] or concept indexing [10] is that the 'importance' of a component can be derived from the weight it receives by an analysis (be it singular value decomposition or k-means clustering, respectively). Correspondingly, we here rank the importance of terms for explaining clustering results based on the weights they have in their cluster centroids.

In order to be able to control how many terms remain to describe the clusters and, hence, be concise, we discretize the term ranks into three descriptions 'very important', 'important', or 'uninteresting' by two thresholds θ_1, θ_2. In our running example, we set θ_1 to 7 % and θ_2 to 20 % of the maximal value. We can then explain the clustering results by considering the terms that are at least 'important' for a resulting cluster.

3.3 Examining BiSec–k–Means Explanations on the Reuters-21578 Dataset

Table 1 shows the highest ranked terms from the centroids of ten (out of 100) resulting clusters on the Reuters-21578 dataset together with their value in the respective centroid. All listed values are above the lower threshold $\theta_1 = 7\%$ (i.e., they are at least 'important'). In general the set of terms that exceed the threshold is much larger (e.g., up to 50 terms of a cluster centroid exceed θ_1) than the set that can be listed here.

A general overview of these results reveals that it is hard to understand the results. While some part of the difficulty stems from the simple, tabular way in which it is presented to the user, quite a substantial part of the difficulty comes from the sheer fact that

Table 1. The highest ranked terms in the first ten out of 100 clusters resulting from a BiSec–k–Means run on the Reuters-21578 dataset (ordered by their values in the respective centroids).

Cluster 0		Cluster 1		Cluster 2		Cluster 3		Cluster 4	
amount	0,12	depository financial instit	0,09	loss	0,34	Irani, Iranian, Persian'	0,14	indebtedness, liability, fin	0,12
billion, one million million,	0,11	financial institution, finan	0,09	failure	0,33	Iran, Islamic Republic of	0,13	obligation	0,12
large integer'	0,11	rate, charge per unit'	0,09	nonaccomplishment, non	0,32	gulf	0,13	debt	0,12
integer, whole number'	0,11	charge	0,09	Connecticut, Nutmeg Sta	0,28	vessel, watercraft'	0,12	written agreement'	0,1
insufficiency, inadequacy	0,1	institution, establishment	0,09	ten, 10, X, tenner, decad	0,24	ship	0,12	agreement, understandin	0,08
deficit, shortage, shortfall	0,1	loss	0,08	American state'	0,23	craft	0,12	creditor	0,08
number	0,09	monetary unit'	0,07	state, province'	0,22	Asian, Asiatic'	0,11	lender, loaner'	0,08
excess, surplus, surplusa	0,09	central, telephone exchar	0,07	system, unit'	0,19	person of color, person o	0,10	statement	0,07
overabundance, overmuc	0,09	financial loss'	0,06	network, net, mesh, mest	0,19	Asian country, Asian nati	0,10	billion, one million million,	0,06
abundance, copiousness	0,09	outgo, expenditure, outla	0,06	September, Sep, Sept'	0,18	oil tanker, oiler, tanker, ta	0,10	large integer'	0,05

Cluster 5		Cluster 6		Cluster 7		Cluster 8		Cluster 9	
text, textual matter'	0,15	loss	0,34	gross sales, gross revenu	0,11	tender, legal tender'	0,15	metric weight unit, weight	0,15
matter	0,15	failure	0,33	sum, sum of money, amo	0,09	offer, offering'	0,14	metric ton, MT, tonne, t'	0,15
letter, missive'	0,15	nonaccomplishment, non	0,32	income	0,09	medium of exchange, mo	0,11	mass unit'	0,14
sign, mark'	0,13	common fraction, simple	0,22	financial gain'	0,09	speech act'	0,1	palm, thenar'	0,14
clue, clew, cue'	0,13	fraction	0,22	gain	0,09	indicator	0,1	area, region'	0,12
purpose, intent, intention	0,11	rational number'	0,22	enterprise	0,05	standard, criterion, meas	0,1	unit of measurement, unit	0,10
evidence	0,11	real number, real'	0,22	business, concern, busin	0,05	reference point, point of r	0,09	organic compound'	0,10
indication, indicant'	0,11	complex number, comple	0,22	assets	0,05	signal, signaling, sign'	0,08	oil	0,10
goal, end'	0,1	one-half, half'	0,22	division	0,05	acquisition	0,06	lipid, lipide, lipoid'	0,10
writing, written material, p	0,07	revolutions per minute, rp	0,22	army unit'	0,05	giant	0,06	compound, chemical com	0,08

there are only few meaningful structures that can be represented to the user at all. To substantiate this claim, let us investigate in detail what kind of structures are available for subsequent explanation in a visualization tool and which are not.

For instance, one may recognize from Table 1 that clusters 2 and 6 are *similar* because they both are about ‘loss’, ‘failure’ and ‘non-accomplishment’. Also, the human observer may *interpret* a cluster description like the one of cluster 1, in order to guess that the list ‘depository financial institution’, ‘financial institution’, ‘rate’, ‘charge’, ‘institution’, ‘loss’, ‘monetary unit’, ‘financial loss’, ‘expenditure’ probably means that this cluster is about financial transaction with loss (which is also correct when investigating the corresponding Reuters news documents). While these structures are not that easy to find in the tables, it is not hard to imagine a user interface to facilitate their discovery.

However, there are meaningful structures that are more difficult to find. For instance, the occurrence of ‘oil’ relates Cluster 3 (ranked further down in the list) with Cluster 9 and several other clusters (from the set of clusters 10 to 99). Along similar lines, it would be nice to see how switching from a general concept like ‘chemical compound’ to a more specific one like ‘oil’ switches the set of associated clusters. Eventually, one would like to find that a particular type of oil or a term like ‘palm’ is a unique property of Cluster 9 as compared against all other clusters. Such structural dependencies require a further analysis as we propose in the following sections.

Eventually, we want to summarize the problems encountered by extracting explaining terms from cluster centroids: This model — if used on its own — assumes that the ranking of terms adequately reflects the importance of terms, which is often not the case (e.g., for Cluster 6 it remains unclear what type of ‘loss’ is encountered). In fact, importance frequently depends on what terms help to explain commonalities between clusters and what terms help to explain differences between clusters — an analysis provided by the next step.

4 Computing the Lattice of Cluster Representations

The clusters obtained by the previous step have the advantage that they cluster similar documents. However, the clustering does not give a description of how the clusters are

related to each other, i.e. an explicit account of what their commonalities and differences are. To derive a lattice that incorporates this account, we exploit Formal Concept Analysis.

4.1 Formal Concept Analysis

Formal Concept Analysis (FCA) was introduced for modeling the concept 'concept'[8] in terms of lattice theory. We recall the basics of FCA as far as needed for this paper. An extensive overview is given in [6]. To allow a mathematical description of concepts as being composed of extensions and intensions, FCA 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*".

A straightforward way of modeling our problem in FCA would be to let the set of objects consist of all clusters determined in the previous step, i. e., $G := \mathcal{C}$ and let the set of attributes consists of all terms which remain from the step described in Section 3.3, i. e., $M := \mathcal{T}_c$. In order to obtain a more fine-grained view, we additionally apply *conceptual scaling*. We use the two thresholds of our example and impose 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: $G := \mathcal{C} \times \{\theta_1, \theta_2\}$, $M := \mathcal{T}_c$, and $((C, \theta_i), t) \in I :\Longleftrightarrow \ (\boldsymbol{t_C})_t \geq \theta_i$. The relation I, applied to a pair (C, θ_i), returns thus the set $\{(C, \theta_i)\}'$ of all attributes which are more or less (i. e., with threshold θ_i) relevant for cluster C.

From a formal context, a concept hierarchy, called *concept lattice*, can then 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,[9] called the *concept lattice* of $\mathbb{K}$ and denoted by $\underline{\mathfrak{B}}(\mathbb{K})$.

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 $\mathbb{K}' := (G', M', I')$ with $G' := \mathcal{D}$, $M' := \mathcal{T}_c$, and $(d, t) \in I'$ iff $d \in C$ and $(\boldsymbol{w}_C)_t \geq \theta$ for some cluster $C \in \mathcal{C}$. This context is an approximation of the descriptions of the documents by term vectors, with the property that all documents in one cluster obtain exactly the same description.

Clustering the objects before applying FCA is an abstraction that might be considered a loss of information. However, it is predominantly beneficial for the following reasons. Firstly, it reduces the number of objects such that FCA becomes more efficient. Secondly, the technique is robust with regard to upcoming documents: A new document is

[8] 'concept' in FCA is a different notion than 'concept' in a thesaurus or ontology.

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

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. Finally and most importantly, formal concept analysis applied directly on all documents suffers from the low co-occurrence of terms. The application of FCA on the Reuters-21578 dataset has shown that hardly any two texts are placed into a common node of the lattice. Thus, the lattice became large, unwieldy and hard to understand for the human user. Therefore, in our approach we first cluster a large number of texts (e.g. 10^5) into a more manageable number of clusters (e.g. 10^2). Only then we compute the lattice, in order to allow for abstracting from some randomness in the joint occurrences of terms.

4.2 The Lattice of the Reuters Clusters

In the Reuters setting, we obtain from the representation computed in the previous section a list of over hundred formal concepts. Each of them groups together clusters of the previous steps. This grouping indicates the conceptual similarity of the clusters. For instance, we obtain a formal concept, which we here refer to by (*), that has {CL 3 (m), CL 9 (m), CL 23 (m), CL 79 (m), CL 85 (m), CL 95 (m)} as extent, and {organic compund, oil, 'lipid, lipide, lipoid', 'compound, chemical compound'} as intent.[10] This formal concept indicates the commonalities ('conceptual similarity' when calling it by FCA terminology) of these clusters: the majority of documents within these clusters are about oil.

The formal concept (*) has three direct subconcepts: the first has {CL 3 (m)} as extent, and the attributes from above plus some attributes like 'oil tanker' and 'Iranian' as intent. The second has {CL 9 (m)} as extent, and the attributes from above plus some attributes like 'area', 'palm', and 'metric ton' as intent. The third subconcept has {CL 23 (m), CL 79 (m), CL 85 (m), CL 95 (m)} as extent, and the attributes from above plus 'substance, matter' as intent. These three subconcepts of (*) show what distinguishes the clusters grouped together in the formal concept (*). The majority of documents in Cluster 3 are about transport of oil (from Iran), those in Cluster 9 about (packaging of) palm oil, and those in the remaining clusters about crude oil.

This example shows that the lattice computed on the resulting clusters can in fact provide meaningful explanations about the commonalities and differences of the set of clusters — beyond what could be provided in Section 3.3. Since it remains inconvenient to figure out these structures just from the list of formal concepts, we furthermore exploit techniques for visualizing the computed lattice in the next step.

5 Visualizing the Concept Lattice

We make use of *Hasse diagrams* for visualizing the concept lattice. They follow the conventions for the visualization of hierarchical concept systems as established in the international standard ISO 704. Figure 1 highlights a part of the concept lattice of our context by a Hasse diagram. It will be explained in detail below. The lattice was computed and

[10] (m) stands here for the important and (h) for the very important terms with the higher threshold values.

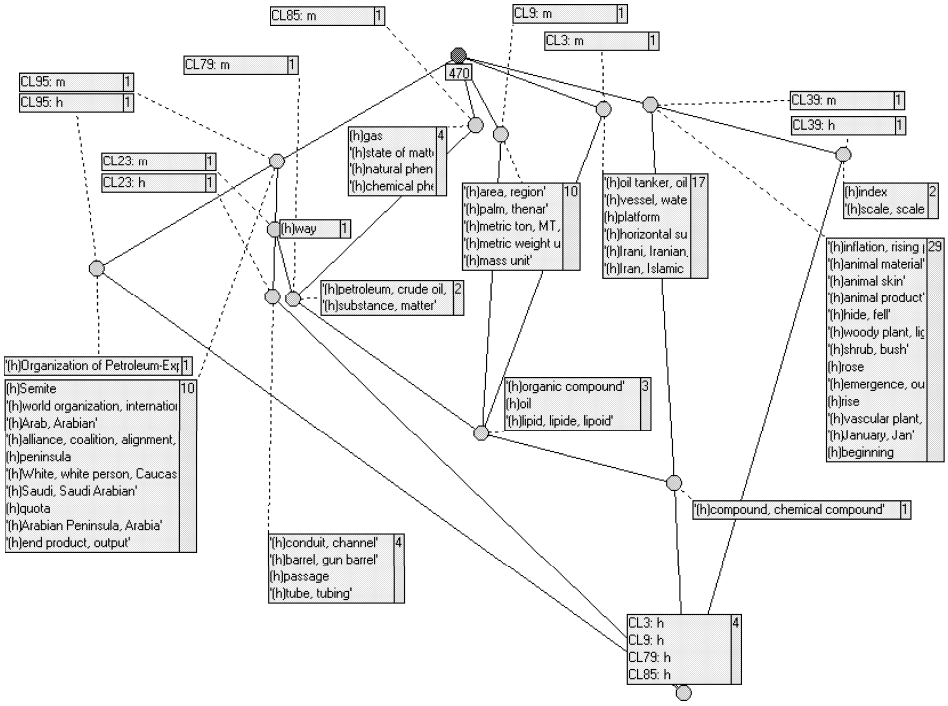

Fig. 1. The resulting conceptual clustering of the text clusters (visualized for the clusters related to chemical compounds.

visualized using the Cernato software of NaviCon Gmbh.[11] It shows all clusters where the value of the synset 'compound, chemical compound' in the centroid is above the threshold $\theta_1 = 7\%$.

In a Hasse diagram, each node represents a formal concept. Due to technical reasons, we reverse the usual reading order: A concept $\mathfrak{c}_1 \in \underline{\mathfrak{B}}(\mathbb{K})$ is a subconcept of a concept $\mathfrak{c}_2 \in \underline{\mathfrak{B}}(\mathbb{K})$ if and only if there is a path of descending(!) edges from the node representing $\mathfrak{c}_1$ to the node representing $\mathfrak{c}_2$.

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.

For example, the concept in the lower middle of the diagram labeled by 'oil' is the concept (*) that we encountered above. In the diagram, we can see that it is part of a chain of concepts with increasing specificity. The most general of them (beside the top concept) contains in its extent clusters of documents addressing chemical compounds

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

(with a medium occurrence) : Clusters 3, 9, 23, 39, 79, 85, and 95. The next concept is the concept (*). Its extent is restricted to those clusters related to oil: all clusters from above beside Cluster 39. We already discussed the subconcepts of (*) above. In the diagram one can see that there are in fact exactly three subconcepts. The one of them in which crude oil is considered, i. e., the one containing the Clusters 23, 79, 85, and 95 in its extent, branches again: While no more information is available about cluster 79, the documents in Cluster 23 and 95 are about the transport and cluster 95 additionally about the oil quotas of the OPEC organization.

Let us also analyze possible problems our approach may encounter: Cluster 85 is also about oil but the intent of the concept is labeled by 'gas'. A closer look to the concepts of the label reveals an additional topic in cluster 85 namely 'gas' as a state of matter which is the first sense in WordNet. An inspection of the documents reveals a mistake of our disambiguation strategy. In the actual documents gas was used as a synonym of gasoline and not as a state of matter. Additionally, some important words are missing in our cluster description which would better explain the content. An important one is 'refinement'. It has a weight that is marginally below the threshold. Thus, we here miss the explanation that Cluster 85 is talking about the refinement of crude oil to gasoline.

To conclude, in the visualization of the concept lattice, we are able to navigate the structures explaining commonalities and differences between different clusters such as manifested in the lattice computed by FCA. The lattice extends the set of meaningful, explanatory structures by means that relate clusters to each other and that exploit the hierarchy defined in the thesaurus for this purpose as a side effect. On the other hand, without BiSec–k–Means as a preceeding step, the FCA step would not have produced a lattice of reasonable and understandable size, because individual texts are too volatile what concerns the joint occurrence of relevant terms. As shown, our approach thus combines the thorough analysis of FCA with the reduction of term and document space to a concise, but relevant basis.

6 Related work

As just summarized, the orginality of our approach is not so much based on the individual algorithms used, as vector representations, BiSec–k–Means, Formal Concept Analysis and Hasse Diagramms are all well known, but on their original integration. This integration serves the purpose to achieve a careful balance with regard to the *granularity* of information used for explanation in three dimensions. First, it automatically finds the adequate level of generalization of concepts in the thesaurus (e.g., 'financial institution' instead of 'bank', whereby only the latter actually appears in the texts). Second, it restricts the term space to a subspace. Thereby, the major components in the cluster centroids are terms that are particularly able to group and discriminate larger text subsets. Third, our approach restricts the document space to a subspace. The subspace abstracts from outlying non-occurrences of individual terms (e.g., one document being about 'financial losses in business acquisitions', but only exhibiting the 'loss' information and not 'company B burned money').

Our experiments have shown, that in order to come up with a concise, but elaborate description, one must carefully balance these dimensions before computing a lattice-based,

hence expressive, explanation. Against this background we may compare paradigms related to our approach.

Hierarchical text clustering — by agglomerative or by partitional algorithms — may be used to derive a tree of cluster representations. One document is in general not only found in one cluster, but it is assigned to a hierarchy of clusters. The hierarchy with its clusters at different levels of representations may be used to describe how clusters are similar or different. Unlike FCA, however, the tree-like hierarchy does not allow multiple assignment of categories, as is common for text documents such as Reuters-21578 news.

We have explored explanations produced by applying common rule/decision tree learners like Ripper [2] or C4.5 [14] on resulting clusters. While the result that these algorithms produce may be very good to classify into the categories they learn, they tend to produce a larger number of rules to explain a single resulting cluster. The explanation for 100 clusters appears to be rather unmanageable for a human user.

We have mentioned that conceptual clustering techniques might be applied directly on the text representations instead of on the text cluster representations. We have also mentioned that there arise problems because of the large term and document space. There are means to reduce the term and document space based on counting support for formal concepts (e.g., Titanic [18]). However, this type of algorithm is rather new and we don't know about work that would have applied it to texts in any way.

Latent Semantic Indexing (LSI; cf. [4, 7]) constitutes a paradigm that groups words into 'concepts' based on their cooccurrences in a given dataset. LSI then allows for text clustering or classification taking into account these 'concepts'. Compared against our approach, its biggest advantage is that a thesaurus is not needed, but the largest disadvantage of LSI is that the notion of 'concept' that LSI introduces cannot easily be explained to a common user. Also, an explanation by a more general concept (the 'first dimension' sketched above) is not possible.

Finally, Karypis and Han [10] have built on the principal idea of reducing term space from LSI, but introduce 'concepts' that are based on the automatic clustering of words. They achieve performance quality comparable to LSI, but their method is more accessible to a human user and might be integrated in the future with a manually defined core thesaurus. We see here a promising line of further research combining their idea of concept construction by clustering (also found in other areas like ontology learning [12]) with its immediate use in text classification and clustering as well as in explaining clustering results as we have proposed in this paper.

7 Conclusion

In this paper, we presented a novel combination of known techniques for text clustering. First, we extended the typical vector space representation of text by synsets of WordNet, in order to exploit its semantics. Then we clustered the documents with the BiSec–k–Means algorithm, using the cosine for measuring the similarity of documents. For each cluster, we extracted a conceptual description, which we used for arranging the clusters in a lattice using Formal Concept Analysis. This blend of known techniques has been shown to combine the benefits of each of the techniques involved: WordNet provides means to identify apparently different terms on a higher level of abstraction; BiSec–k–Means structures the domain and reduces it to a manageable size; the extracted cluster descriptions

helps identifying the content of individual clusters; and FCA and its visualization means show up the relation between those clusters.

References

1. C. Buckley and A. Lewit. Optimizations of inverted vector searches. In *SIGIR-1985*, pages 97–110, 1985.
2. William W. Cohen. Fast effective rule induction. In Armand Prieditis and Stuart Russell, editors, *Proc. of the 12th International Conference on Machine Learning*, pages 115–123, Tahoe City, CA, July 9–12, 1995. Morgan Kaufmann.
3. D.R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *SIGIR-1992*, pages 318–329, 1992.
4. 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.
5. C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.
6. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin–Heidelberg, 1999.
7. T. Hofmann. Probabilistic latent semantic indexing. In *Research and Development in Information Retrieval*, pages 50–57, 1999.
8. A. Hotho, A. Maedche, S. Staab, and R. Studer. SEAL-II — the soft spot between richly structured and unstructured knowledge. *Journal of Universal Computer Science (J.UCS)*, 7(7):566–590, 2001.
9. A. Hotho, S. Staab, and G. Stumme. Wordnet improves text document clustering. In *submitted*, 2003.
10. George Karypis and Eui-Hong Han. Fast supervised dimensionality reduction algorithm with applications to document categorization and retrieval. In *Proceedings of CIKM-00*, pages 12–19. ACM Press, New York, US, 2000.
11. G. Kowalski. *Information Retrieval systems-theory and implementations*. Kluwer Academic Publishers, 1997.
12. A. Maedche and S. Staab. Ontology learning for the semantic web. *IEEE Intelligent Systems*, 16(2):72 –79, 2001.
13. D. Mladenic. Text learning and related intelligent agents. *IEEE Expert*, July/August 1999.
14. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, California, 1993.
15. C. Van Rijsbergen. *Information Retrieval*. Buttersworth, London, 1989.
16. G. Salton. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley, 1989.
17. M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 2000.
18. G. Stumme, R. Taouil, Y. Bastide, N. Pasqier, and L. Lakhal. Computing iceberg concept lattices with Titanic. *J. on Knowledge and Data Engineering*, 42:189–222, 2002.
19. O. Zamir, O. Etzioni, O. Madani, and R.M. Karp. Fast and intuitive clustering of web documents. In *KDD-1997*, pages 287– 290, 1997.