Generating Concept Ontologies Through Text Mining

Lipika Dey[1]   Ashish Chandra Rastogi   Sachin Kumar
Department of Mathematics
Indian Institute of Technology, Delhi
Hauz Khas, New Delhi - 110016

Abstract

Designing mechanisms for creating concept ontologies automatically is an important research problem. In this work we have proposed a rough-set based mechanism to generate concept ontologies with concepts mined from documents. When the concept ontology is mined from preclassified documents, the output signifies the core set of domain concepts and their inter-relationships that define the categories, as well as the inter-category relationships. When the ontology is mined from a heterogeneous collection, the documents are first clustered into homogeneous groups and then mined for concepts. Rough set based lower and upper approximations have been used to identify core concepts and associated concepts for a domain or a group. The scheme has been tested over multiple domains.

1. Introduction

With the proliferation of the Internet, the primary media of information-exchange today is the text document. However the unstructured nature of text makes text information retrieval a non-trivial task. Information sharing across applications is only possible when they subscribe to the same vocabulary. Ontologies help in knowledge management by providing a formalization for representing domain-specific knowledge as well as reason with it. Documents and other application-specific artifacts are semantically annotated and accessed using the domain vocabulary provided by the ontology. While an isolated term in an unstructured natural language text tends to be ambiguous, when placed within an ontological structure, the ambiguity of the term can be resolved with the aid of other related concepts. However, building such ontologies require considerable amount of expertise and effort, and this becomes a bottleneck in designing ontology-based applications. The need for automated procedures to create, maintain and use ontologies can hardly be over-emphasized. An approach to generate relevant ontologies is to mine these concepts from documents belonging to the domain itself.

In this paper, we present a rough-set based concept hierarchy generation mechanism. Given a pre-classified collection of documents, the proposed scheme mines the collection to generate concept-based descriptions for these categories. These descriptions also highlight the inter-relationships of the categories themselves. This is akin to auto-emergence of the domain ontologies, given a collection of representative documents. On the other hand, given a non-classified collection of documents, the first step is to group them into homogeneous collections and then mine the collections for extracting concepts that can best describe these groups. This is akin to self-genesis or boot-strapping of a core-set of concepts for an application. The emerging concept hierarchy enables unsupervised organization of documents. The proposed framework opts for a tolerance rough set based document enrichment mechanism, to identify related concepts. A rough approximation based technique is thereafter applied to generate concept maps. The rough set based approach facilitates representation of domain knowledge at multiple levels of granularity. The concept map provides insight into the way the documents are grouped and also provides a mechanism for conceptual indexing of the documents.

2. Related Work

Mulholand et al.[3] proposed ENRICH, a methodology that promoted organizational learning within an enterprise by providing support for a number of learning processes. Two approaches were mainly used for ontology learning. In the set theoretic approach, word co-occurrence or distributional hypothesis was used as a measure for analysing the text data from which the ontology was built. The other approach was to use the WordNet to determine the term association graph for patterns mined from the documents. OntoKnowledge is a project in which a complete set of tools have been built to generate ontologies from annotated documents.

Velardi et al.[11] specify a three-step process to develop concept ontologies from text documents. The three step process comprises concept learning, concept validation and ontology management. Starting with WordNet as an initial ontology, this system applies a set of semantic rules to disambiguate senses of co-occurring words and build the final ontology. It also lays emphasis on deriving consensus among domain experts to manage the ontology.

Clustering algorithms are aimed at organizing the documents into homogeneous collections in an unsupervised way. It has various applications. One of its uses have been to route queries efficiently over the

[1] The author is currently on leave from I.I.T., Delhi and working with Webaroo Technologies India Pvt. Ltd.

Proceedings of the 2006 IEEE/WIC/ACM International Conference
on Web Intelligence (WI 2006 Main Conference Proceedings)(WI'06)
0-7695-2747-7/06 $20.00 © 2006 IEEE

distributed web structure, by directing user queries to appropriate servers, where a relevant cluster is hosted. The partitional clustering algorithm K-means and its variations is the most commonly used technique for document clustering.

Dhillon and Modha [4] used the spherical k-means algorithm to cluster a set of document vectors. The algorithm outputs k-disjoint clusters each with a concept vector that is the centroid of the cluster normalized to have unit Euclidean norm. Zhao and Karypis [10] reported a comparative study on the performance of agglomerative and partitional clustering algorithms for documents. Kogan et al. [5] proposed a hybrid clustering scheme that uses singular value decomposition (SVD) followed by a k-means type partitioning algorithm for text mining. Ngo and Nguyen [6] proposed the use of a Tolerance Rought Set Model(TRSM) for clustering the search results. Vivisimo[2] also uses clustering as a technique to divide the documents returned as a result of search into coherent groups, for the convenience of users.

Rigutini and Maggini [8] proposed a semi-supervised document clustering algorithm based on Expectation Maximization. Supervision is provided as a set of initial groups of documents which should be together. Basu et al. [1] proposed a probabilistic model for semi-supervised clustering based on Hidden Markov Random Fields. The model combines constraints and Euclidean distance learning, and allows the use of a broad range of clustering distortion measures like Bregman divergences and directional similarity measures.

Hierarchical clustering on the other hand can provide a better insight into document clusters by allowing visibility at multiple levels of granularity. Zhao and Karypis [9] have proposed constrained agglomerative clustering algorithms that combine the features of both partitional and agglomerative clustering. These algorithms build meaningful hierarchies out of large document collections as a means of providing tools for interactive visualization at different levels of granularity.

3. Basics of Rough Set Theory

In this section we give a very brief overview of the rough-set concepts that we have used in this paper. Rough sets, introduced in [7] can be used to represent ambiguity, vagueness and general uncertainty. A rough set is an imprecise representation of a crisp set in terms of two subsets, a lower approximation and an upper approximation.

The central point of rough set theory is the notion of set approximation based on the indiscernibility relation, *R,* defined over the set of objects belonging to the universe U. *R* is an equivalence relation, and induces a complete partitioning of the universe into equivalence classes, where each equivalence class consists of objects that are indiscernible from each other. The lower and upper approximations of a set *X*, represented by $L_R(X)$ and $U_R(X)$ respectively, are then defined as follows:

$L_R(X) = \{x \mid x \epsilon U$ and $[x]_R$ is a proper subset of $X\}$ and

$U_R(X) = \{x \mid x \epsilon U$ and $[x]_R$? X ? Φ $\}$ where $[x]_R$ denotes the equivalence class of x, induced by *R*.

Together, the pair $(L_R; U_R)$ constitutes a rough approximation (or rough set) of concept X, where $L_R(X)$ consists of those objects that definitely belong to X, and $U_R(X)$ consists of those objects that possibly belong to X. In text retrieval, the synonymy relation being an equivalence relation can provide the basis for partitioning a collection of words into equivalence groups. However, one of the more commonly used relations for text retrieval is the word co-occurrence relation, which is not an equivalence relation.

A Tolerance Rough Set Model (TRSM) for text retrieval problems was proposed in [6]. A relation *R* is said to be a *tolerance relation* if it is reflexive and symmetric. Word co-occurrence is obviously a tolerance relation. For each object $x \epsilon U$ the tolerance class of x is denoted by I(x) and is defined as the collection of all those objects which are related to x by the underlying tolerance relation *R*. Thus, $I(x) = \{ y \mid y \epsilon U$ and yRx where R is a tolerance relation$\}$.

Lower and upper approximations over the tolerance rough space is defined using a *structurality function* P:I(U)?{0,1}. $P(I(x)) = 1$ if $|I(x)| > \alpha$, where α is a user given parameter, else $P(I(x)) = 0$. $L_R(X)$ and $U_R(X)$ for a set X are now defined as follows:

$L_R(X) = \{x \mid x \epsilon U$ and $P(I(x)) = 1$ and $\upsilon(I(x), X) = 1\}$

$U_R(X) = \{x \mid x \epsilon U$ and $P(I(x)) = 1$ and $\upsilon(I(x), X) > 0\}$, where $\upsilon(Y, X) =$ |Y?X|/|X|, is called the *vague inclusion function*. Thus the lower approximation of X now consists of all those elements whose tolerance classes are wholly contained in X. The upper approximation of X consists of those elements whose tolerance classes have a non-null intersection with X.

4. The concept hierarchy

Before going into the details of generating a concept hierarchy, we first present an overview of the proposed structure. As mentioned earlier, when a set of pre-classified documents are processed, the aim of concept hierarchy generation is to extract the set of concepts from the documents that can represent the class definitions well enough. Since word co-occurrence can capture context better than isolated words, hence we use the term co-occurrence matrix as a starting point for content analysis. Using the term co-occurrence frequencies in a set of documents, a document is first *enriched* to contain words which frequently co-occur with terms contained in it, even if those terms are not present in the document itself. Thus if a document contains a concept C1, and the concept C1 is found to co-occur with concept C2 *quite often* in the corpus, then even if the document does not contain the concept C2, it may be inferred that the contents of the document cover the concept C2. Co-occurrence is not a

[2] http://vivisimo.com/html/whyclustering

transitive relation, but a tolerance relation. The enrichment of a document is computed using a Tolerance Rough Set Mechanism (TRSM). Enriched documents are represented as term-weight vectors, whose computation is elucidated in section 5. Using enrichment, a collection of terms are identified that can collectively represent a category. This increases the granularity of the term-based representation scheme.

Generating the concept hierarchy for a set of pre-classified documents, entails finding a set of terms and their relationships that can best define the categories. The *lower approximation* of a category defines a set of core terms that can be considered as essential to describe the category. The *upper approximation* of a category defines a set of terms that are associated to the core terms and are well represented in the category. The concept hierarchy is represented as a tree, where each node can be viewed as a generalization of its children. The lowest level nodes in a concept hierarchy are called the *units* while the higher-level nodes are called *composite collections.* A composite collection contains the union of the document collections belonging to the constituent units.

Each concept node is represented by its *lower* and *upper* approximations. The lower approximation of a concept node comprises concepts that are present in all documents grouped under that node. The upper approximation represents set of concepts that are well-represented in the corresponding group. The lower and upper approximations of composite collections are computed using lower and upper approximations of *unit* collections. The concept hierarchy generation mechanism also extracts the taxonomic and partonomic relations among concepts. Thus terms in the upper approximation of a node are generalizations of the terms in the corresponding lower approximation. The terms in a composite collection are related to the terms in the corresponding units through the part-whole relationship.

Given a heterogeneous collection of documents, the ontology is allowed to emerge from the collection as one which can help in describing homogeneous sub-groups. A document can be a member of multiple groups at the same time, depending on the underlying clustering algorithm used. We have considered a partitional clustering algorithm in which each document is associated to a unique cluster. These clusters are thereafter arranged into a hierarchical organization where higher-level clusters are formed by combining lower level clusters.

For example, while working with complaint-related documents collected from different web-sites, it was found that unit clusters could describe complaints related to a particular company, while higher level clusters could be related to concepts describing products, and an even higher level could be associated to the nature of complaints. In this case, concepts related to unit clusters were found to be company names like “Sony”, “Compaq”, “Nokia”, “LG” etc., while at second level the concepts were “computer”, “cell phones”, etc. along with the company names, and third level concepts extracted were terms like “screen”, “monitor”, “power”, etc. which represented nature of complaints. It is inferred that the unit level clusters consisted of documents belonging to the same company. The first level composite clusters consisted of complaints related to different companies but same products, while the second level composite clusters consisted of documents pertaining to similar types of complaints for different products and different companies.

The concept hierarchy has several applications. Given a collection of documents, the hierarchy can provide a good indexing mechanism for the collection. To carry on with our earlier example, the hierarchy generated from complaint documents can be used by the companies to assign tasks to individual specialized groups. The terms in the concept nodes can be used to identify required expertise while dealing with a group of complaints. Subsequent sections contain functional details about document enrichment, concept hierarchy generation and the clustering mechanism employed. Sample hierarchies generated have been presented as examples to illustrate the effectiveness of the mechanism.

5. Document Enrichment

Let D = $\{D_1, D_2, ..., D_n\}$ represent a collection of n documents. A document D_i is represented using the standard Vector Space Representation: $d_i = \{t_1{:}w_{1i}, t_2{:}w_{2i},...,t_{k:}w_{ki}\}$, where k is the total number of unique words(terms) in the document collection and w_{ki} represents the weight of term t_k in $d_{i.}$ Initially for each document, w_{ki} is computed using the TF-IDF approach, where $w_{ki}=f_{ki}*\log(n/v)$, where f_{ki} denotes the frequency of term t_k in D_i and v denotes the total number of documents in which t_k occurs with positive frequency. All weights are normalized. The enrichment of a document D_i is denoted by Ed_i and is then generated as follows.

For each term $t_{i,}$ let $f_D(t_i, t_j)$ denote the number of documents in which t_i co-occurs with t_j. Using TRSM the tolerance class of t_i is defined as $I_\theta(t_i)=\{t_j|f_D(t_i, t_j)>\theta\}$, where θ is a given threshold. Thus the tolerance class of each term contains the set of terms with which it frequently co-occurs in the corpus. Each term is included in its own tolerance class. The enriched document is obtained by adding to it the new terms occurring in the union of tolerance classes of all terms it originally contained, and which have frequency greater than θ. Thus an enriched document is denoted by $d^e{}_i = \{I_\theta(t_m)|f_{mi} > \theta\}$.

The weights for the terms in the enriched document are computed as follows. For each original term in the document the weight is re-computed using the earlier formula, but the frequency values are now computed over the enriched document collection. The weight assigned to a newly added term e_j is computed as $(f_j/n)*\log(1+ f_j/n)$, where f_j is the frequency of the term e_j in the enriched document collection. This function, which is a variation of the original TF-IDF formula is chosen to ensure that the weight of the added terms do not exceed that weight of the original terms. An enriched document vector is now represented by $d_i = \{t_1{:}w^e{}_{1i}, t_2{:}w^e{}_{2i},...,t_{k:}w^e{}_{ki}\}$, where $w^e{}_{ki}$

represents the new weight of term t_k in d_i. All weights are normalized to the range 0-1.

During enrichment of pre-classified documents, the co-occurrence matrix is constructed from terms occurring in the documents of the same class. Given a heterogeneous collection, the enrichment is done prior to clustering with the co-occurrence matrix constructed form the whole collection.

6. Generating the Concept Hierarchy

Let us suppose that the document collection contains documents from r categories or r clusters. The center of a category or a cluster is defined as a collection of terms that belong to all enriched documents of the category or cluster respectively. Let 2^r denote the power set defined by these r elements. The concept hierarchy is defined as a *concept tree* over this collection, where each node n_i satisfies the following property: (i) it is an element of 2^r and (ii) if n_1 is a child of n_2 then n_1 is a subset of n_2. At the leaf level of this tree lies the r initial categories(clusters), which are called the unit categories (clusters). Figure 1 shows the structure of a concept hierarchy that is built from three unit categories (clusters). The concept hierarchy defined thus generates multiple concept representations for the same collection, by considering the collection in isolation and also in combination with other homogeneous collections. Thus a set of documents S1 can be covered by a set of concepts represented by C1 at one level, while S1 along with another set S2 may be covered by another set of concepts C2. Hence the hierarchy is capable of assigning labels to a collection from multiple perspectives.

Each node in concept hierarchy represents a set of concepts which are extracted using the concepts of *lower approximation* and *upper approximation.* The lower approximation of a node denotes a set of concepts that are *definitely* present in *all* documents associated to the node. The upper approximation of a node consists of a set of concepts that are definitely present in *some* documents associated to it. Thus the lower approximation represents a set of concepts that the node *definitely* represents. The upper approximation represents a set of concepts that the node *possibly* represents. Concept approximation is initiated at the leaf level nodes which represent the original clusters and propagated up the concept tree.

Let $C_1, C_2, ..., C_r$ denote the centers of the unit categories (clusters). Let t_j denote a word that is present in all documents belonging to the category (cluster) j with non-zero weight. Let $sup(t_j)$ denote the maximum, $inf(t_j)$ denote the minimum and $avj(t_j)$ denote the average weights of the word t_j in cluster j. The lower approximation L_j of category (cluster) j is computed as follows:

$L_j = \{t_j \mid t_j \epsilon C_j$ and $avj(t_j)$? $min_h(sup(t_h))$, where 1? h? $|C_j|$).

The computation of the lower approximation ensures that it contains only those concepts that are present in all the documents with a sufficiently high weight.

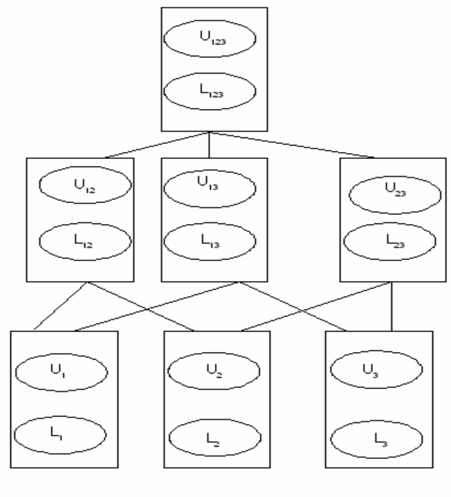

Figure 1. A sample concept hierarchy

To compute the upper approximation of a category (cluster), all terms belonging to the documents of the category (cluster) are considered. Let T_j denote the union of all terms that occur in majority of the documents. To obtain this set of words, the weight vector of each word is considered and all those words which have non-zero weights in at least 50% of the documents are selected as candidate concepts for the upper approximation. The upper approximation U_j of category (cluster) j is computed as follows:

$U_j = \{t_j \mid t_j \epsilon T_j$ and $avg(t_j)$? $max_h(inf(t_h))$, where 1? h? $|T_j|$).

The computation of the upper approximation ensures the inclusion of all those concepts which are reasonably well-represented in majority of the documents in the category (cluster), even though not present in all the documents. The lower and upper approximations of the other nodes in the concept tree are similarly computed using the intersections and unions of the documents belonging to the participating clusters. Thus for the node covering the union of documents belonging to category (cluster) i and category (cluster) j, the lower and upper approximations are computed as :

$L_{ij} = \{t \mid t \epsilon L_i$?L_j and avj((t) ? $min_h(sup(t))$, where 1? h? $|L_i$?$L_j|$) and

$U_{ij} = \{t \mid t \epsilon U_i \mathbf{U} U_j$ and avg((t) ? $max_h(inf(t))$, where 1? h? $|U_i \mathbf{U} U_j|$)

7. A Sample Concept Hierarchy for 20 News Group Documents

We now present sample concept hierarchies generated from the 20NewsGroup collection that contains pre-classified articles categorized into 20 different categories. The concepts have been extracted from 100 documents in each category. Table 1 presents some concepts that were extracted as lower and upper approximations for the respective categories. The concepts which are part of the lower approximation are obviously a part of the upper approximation, and has not been repeated in the table. The concepts extracted are clearly very relevant to a particular category.

Table 1. Concepts extracted for category

Category	Core concepts (lower approximation)	Related concepts (upper approximation)
Politics guns	news,kill,group,firearms,violent,rocket,company, investors,stratus meyers	House, arms, weapons,defense , nation, criminal, violent, burns, religious, assault, survivors,
Politics.mideast	Expansion, terror armenians, extermination, turkish, jake, bony refuse, report bush, carter, employ muslims, jews, population, target civilians, israel, french war, reason, arabs move, peace	Sera, zuma, serdar, race, orion, time, clock, history,forgotten publish, public, import, grant escape, turks, soviet, attempt, believe, states, possible, assert matter, refuse, words, report, friends, list, employment, president, kill, independence, republic, participation, men, resist, targets, islam, civilians, murder, palestinian, policy, respond, bosnia, fight, final, effect, greek, turkey, region, revolution, homeland ..
Religion.christian	geneva, love, believe, scripture, prophecy, heaven, worship, christ eternity, church sin, death doctrine, christian	University, jesus, turn, athos, children, lord, spiritual, divine god, knowledge, group, marriage, commit, concept, words, judge, mistakes, worship, rule, life, born, light, develop, revel, teachings, bible, religion, soul, create, mind, age, reform
Religion.misc	Concept, bible,believe love, god, religion christians, import morality, jewish, jesus sins, kind, christ christian, revel, sin christianity, exist mark, life	(same as core concepts)
Politics.miscellaneous	Study, gay, double, cramer, homosexuals child, members partners, men	Interest, straight clinton, fire, double report,compound black, agents media, uiuc, uchicago congress, charge support, hallam, press reason, promiscuous, group evil, history, murder

It is also observed that there are semantic relations between words in the lower and upper approximations. For example, firearms are a type of weapon, or Bush and Clinton are presidents. Explicit relation extraction requires the active use of a thesaurus.

8. A link-Based Clustering Algorithm for Clustering Heterogeneous Documents

The similarity of two enriched documents d_i and d_j, is denoted by $\xi(d_i,d_j)$ and is computed as the inner product of the enriched document vectors d_i and d_j. All similarity values are also normalized to lie within the range of 0-1. For each pair of documents d_i and d_j the similarity value $\xi(d_i,d_j)$ is used to determine whether the two documents should belong to the same final cluster. Higher the value of $\xi(d_i,d_j)$, higher the chances that they will belong to the same cluster and vice-versa. This information is computed and encoded as the kind of link that is associated to the document pair d_i and $d_{j.}$

Let S_{max} denote the maximum similarity value for the enriched document collection. Based on S_{max} and $\xi(d_i,d_j)$, the link between a pair of documents d_i and d_j is assigned to the *must-link, can-link,* or *cannot-link* categories as follows:

If αS_{max} ? $\xi(d_i,d_j)$? 1 then the documents d_i and d_j form a *must-link* pair.

If βS_{max} ? $\xi(d_i,d_j)$? αS_{max} then the documents d_i and d_j form a *can-link* pair.

If 0? $\xi(d_i,d_j)$? βS_{max} then the documents d_i and d_j form a *cannot-link* pair, where $0<\beta<\alpha<1$.

α and β are a pair of cut-off values that are domain-dependent and controlled by the user to influence the quality of grouping. A high value of α implies a high degree of homogeneity in the groups. A high value of β implies that documents with low values of similarity will be considered as cannot-link pairs.

Experiments show that for grouping technical abstracts α and β values can lie approximately close to 0.9 and 0.3 respectively, which are very coherent in use of significant terms. For complaint related documents these values come down to 0.7 and 0.2 respectively. For handling news group articles as in the 20NewsGroup collection3, these values had to be lowered to 0.6 and 0.05 respectively, to get correct link associations. On analysis of these documents, it was found that some of these documents are very bad samples.

The clustering process makes use of the link information that is generated by the earlier step. It is an iterative algorithm, in which each iteration goes through two phases – *cluster growing* and *cluster pruning*. Initially, the must-link pairs from the document collection are analyzed for identifying the connected components. Each connected component is assumed to be an initial cluster center. When a document is not found close enough to any of the initial clusters, new clusters are grown. At the end of an iteration, each cluster is checked for homogeneity, and if needed documents are removed from a cluster. The clusters retained at the end of an iteration serve as the initial clusters for the next iteration. The process continues till all documents converge to the best possible cluster. The *center* of a cluster is defined by the set of words that belong to the intersection of all documents.

The steps in the clustering algorithm are explained below.

Step 1: Mark all documents as UNASSIGNED.

Step 2: Identify all connected components from the enriched document collection to act as initial clusters.

Repeat the following steps

Step 3: Label each document that is part of a cluster as ASSIGNED.

Step 4: Compute the current cluster centers. Let C^i_1, C^i_2,...,C^i_t denote the centers of the currently existing t number of clusters.

Step 5 Cluster Growing phase: We know that each enriched document that is not a part of any cluster, is related to the

[3] http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naivebayes/20_newsgroups.tar.gz

documents in the cluster either by a *can*-link or a *cannot*-link. For a document d_i, let P_k denote the set of documents of cluster number k with which d_i is connected with can-links, and N_k denote the set of documents of the same cluster with which d_i is connected with cannot-links. The proximity of a document from an existing cluster k is denoted by δ_k and is computed as follows:

$\delta_k(d_i)=|P_k|*\Sigma_{m\in P_k}\xi(d_i,d_m) \quad - \quad |N_k|*\Sigma_{m\in N_k}\xi(d_i,d_m)$. The proximity function thus takes into account both the number of links of each category and the strength of each link. It yields negative values if the cannot-link connectivities override the can-link connectivities. The proximity value of a document to various clusters is used to determine the best cluster for it.

For each document d_i that is marked UNASSIGNED

(i) Compute its proximity to all existing clusters.

(ii) If proximity of d_i comes out to be negative for all existing clusters, then a new cluster C^i_{t+1} is created with d_i.

(iii) Otherwise: Let $\delta_p(d_i)$ be the maximum proximity value attained for d_i.

(iv) If d_i?C^i_p ? NULL, then d_i is added to cluster number p.

(v) otherwise d_i forms a new cluster.

(vi). Label d_i as ASSIGNED.

Step 5: Cluster Pruning phase: During this phase, each cluster, which has more than one document, is checked for homogeneity. For each document d_i assigned to cluster number k, $\delta_k(d_i)$is recomputed taking into account all documents that are currently assigned to the cluster. All documents for which $\delta_k(d_i)$ turns out to be negative are removed from the clusters and are labeled as UNASSIGNED.

Step 6: All clusters that have only one document are eliminated and these documents are marked as “UNASSIGNED”.

Until no change is observed in two consecutive iterations

Step 7: Each UNASSIGNED document is included as a cluster with only one element.

End

The above algorithm (i) eliminate clusters with single documents unless unavoidable, (ii) maintains high cluster homogeneity by ensuring that the cluster center is not null, (iii) pruning off documents which tend to decrease the overall cluster homogeneity. The definition of the center ensures that a set of documents with no overlapping concepts do not form a cluster. While cluster growing emphasizes on utilizing can-link information about a document, cluster pruning emphasizes on using the cannot-link information for meaningful clustering. In most of the practical situations, the algorithm generates clusters with more than one element in each cluster.

9. Sample Concept Hierarchies Built from Heterogeneous Collections

We now present some results to illustrate the working of the entire scheme. Though experiments have been conducted over multiple domains, due to lack of space, we have illustrated only a small concept hierarchy. The first set of documents are picked up from the 20NewsGroup collection, which contains around 20000 articles classified into various categories. The other domain that we show results from are a set of complaint documents that we downloaded from the Internet randomly. This domain is particularly interesting since there is no predefined category or concept map existing for this collection. Hence it is interesting to watch the concept map emerge.

Table 2. Clustering results

Domain	No. of docs.	Accuracy
Medline abstracts[4]	80	80%
20NewsGroup	25	90%
Complaint	30	93.3%
Complaint	60	87%

Table 2 provides the performance of the clustering process for various subsets of documents picked up from three domains. Table 3 provides a summary of 15 complaint documents, which were divided into three clusters. The cluster compositions and the key concepts in the concept hierarchy generated thereof are shown in Table 4. The documents have been picked up for five different companies, with overlapping set of products. The lower approximation of each unit cluster contains company specific exclusive concepts like company names and product names, if the product is exclusive. For example, since Nokia has only cell-phones, all complaints were related to that product and hence the product name is in the lower approximation. However, since Sony has multiple products like laptop computers, TV, music system etc., the complaints were about divergent products and hence the product names do not occur in the set of core concepts. The product names occur in the upper approximation as related concepts. Some composite collections were found to be very interesting. There were complaints for DVDs for both companies Sony and Panasonic. Since HP and Compaq has collaboration on laptops, Compaq occurs as an associated concept for HP also. The upper approximation of a set obviously includes the terms in the lower approximation also, though not explicitly shown in the table. The concepts associated to only some of the more interesting concept nodes of the hierarchy is shown.

Table 3. Description of complaint documents

Group(Documents)	Complaint products
Group 1 (1-5)	Sony TV, Laptop and Sony computer screens

[4] http://www-tsujii.is.s.u-tokyo.ac.jp/~genia/topics/Corpus/

Group(Documents)	Complaint products
Group 2 (6-10)	Nokia phones
Group 3 (11-15)	Compaq presario laptop and computers
Group 4 (16-20)	HP computer and printer
Group 5 (21-25)	Panasonic TV, audio player, VCR-DVD combo

Table 4. Concept hierarchy description

Node Id	Concept approximations
Group 1(Sony)	**Lower approximation** Sony, buy, repair **Upper approximation** manual, press, warranty, sound, lamp, dvd, block, period, tivo, company
Group 2(Nokia)	**Lower approximation** nokia, custom, mobile, phones, repair **Upper approximation** **cell, buttons, send, receive, messages, calls, color, black, grey, warranty, residue, liquid, board**
Group 3 (Compaq)	**Lower approximation** compaq, computer **Upper approximation** laptop, screen
Group 4 (HP)	**Lower approximation** purchase, hp, computer **Upper approximation** purchase, repair, yahoo, problems, refund, replace, hp, compaq, company, hewlett, packard
Group 5 (Panasonic)	**Lower approximation** panasonic, warranty, repair, month, company, purchase, really, refund, regard, problems, full, expect, apparently, kind, expensive, look, forward **Upper approximation** software, fraud, business, customer, fine
1+2	**Nil**
1+3	**upper approximation** Computers
Group1+5	**upper approximation** dvd, sony, panasonic, warranty, repair, screen
Group 2 + 3	Nil
Group 3+4	**Lower approximation** Compaq, computers

10 Conclusions

In this paper we have presented a rough-set based method for grouping a set of documents into a concept hierarchy. Using a tolerance rough set based model, the documents are initially enriched by including additional terms that belong to the document's tolerance space. For a pre-classified collection of documents, the enrichment process is applied over each category. Concepts are extracted for each category. For heterogeneous collections, the enriched documents are first clustered using a two-phase iterative clustering algorithm. Finally the clusters are arranged to form a concept hierarchy, where each node in the hierarchy is represented by a set of concepts that covers a collection of documents. Each node is approximated by two sets of concepts. The lower approximation of a collection of documents represents a set of concepts that the documents definitely cover. The upper approximation of the collection represents a set of concepts that are possibly covered by the collection.

The proposed mechanism has been tested for various domains and found to generate interesting concept hierarchies. The mechanism is presently being used to generate concept hierarchies over medical abstract collections. The concept approximations can be then used to index a collection effectively to answer concept based queries. The proposed mechanism is also ideally suited to generate new domain ontologies, when applied over a representative set of documents from the domain. Further work using a thesaurus to extract the word relations is in progress.

References

[1] S. Basu, M. Bilenko, and R. J. Mooney, A Probabilistic Framework for Semi-supervised Clustering, Proceedings of the 10th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD-2004), Seattle, WA.

[2] J. Davies, D. Fensel and F. van Harmelen (eds), The Semantic Web: Ontology Driven Knowledge Management, John Wiley, 2002.

[3] P. Mulholland, Z. Zdrahal, J. Domingue, M. Hatala, A Methodological Approach to Supporting Organizational Learning, Int'l Journal of Human-Computer Studies 55, 2001.

[4] I.S. Dhillon and D.S. Modha, Concept Decompositions for Large Sparse Text Data using Clustering, Machine Learning, 42(1), pp. 143-175.

[5] J. Kogan, C. Nicholas, and V. Volkovich, "Text Mining with Hybrid Clustering Schemes", Workshop on Text Mining, Third SIAM International Conference on Data Mining (SDM 2003), pp. 5-16.

[6] C. L. Ngo and H. S. Nguyen, "A Tolerance Rough Set Approach to Clustering Web Search Results", in LNCS-3202/2004, PKDD-2004, pp. 515.

[7] Z. Pawlak, Rough sets, Int'l Journal of Computer and Information Sciences, 11, pp. 341-356.

[8] L. Rigutini and M.Maggini, A Semi-Supervised Document Clustering Algorithm Based on EM, Proceedings of Web Intelligence, WI-2005.

[9] Y. Zhao and G.Karypis, Comparison of Agglomerative and Partitional Document Clustering Algorithms, Proceedings of the Workshop on Clustering High Dimensional Data and its Applications, Second SIAM International Conference on Data Mining, pp. 83-93.

[10] Y. Zhao and G. Karypis, Hierarchical Clustering Algorithms for Document Datasets, Data Mining and Knowledge Discovery, 10, pp. 141-168.

[11] P. Velardi, M. Missikoff, P. Fabriani, Using Text Processing Techniques to Automatically Enrich a Domain Ontology, Proceedings of ACM Conference on Formal Ontologies and Information Systems, Ogunquit, Maine, October, 2002.