Hierarchical Document Clustering Based on Tolerance Rough Set Model

Saori Kawasaki, Ngoc Binh Nguyen, and Tu Bao Ho

Japan Advanced Institute of Science and Technology
Tatsunokuchi, Ishikawa, 923-1292 JAPAN
{skawasa,binh,bao}@jaist.ac.jp

Abstract. Clustering is a powerful tool for knowledge discovery in text collections. The quality of document clustering depends not only on clustering algorithms but also on document representation models. We develop a hierarchical document clustering algorithm based on a tolerance rough set model (TRSM) for representing documents, which offers a way of considering semantics relatedness between documents. The results of validation and evaluation of this method suggest that this clustering algorithm can be well adapted to text mining.

1 Introduction

We introduce a document representation model, namely *tolerance rough set model* (TRSM). Rough set theory, a mathematical tool to deal with vagueness and uncertainty introduced by Pawlak [3], has been successful in many applications, in particular in data mining [4]. TRSM employs a tolerance relation instead of an equivalence relation in the original rough set model (see also [5]). We develop a TRSM hierarchical clustering algorithm for documents that exploits semantics relatedness between documents. The algorithm consists of two phases of making representation of each document using TRSM (section 2) and grouping documents by an agglomerative clustering with their approximations (section 3). We report our experiments on five test collections in section 4.

2 A Tolerance Rough Set Model for Documents

Denote the set of M full text documents by $\mathcal{D}$, and the set of N terms from $\mathcal{D}$ by $\mathcal{T}$. Each full text document $d_j \in \mathcal{D}$ is mapped into a list of terms t_i weighted by their importance in the document, as $d_j = (t_{1j}, w_{1j}; t_{2j}, w_{2j}; \ldots; t_{rj}, w_{rj})$ with $w_{ij} \in [0,1]$. Then the TRSM is used for enriching the document representation in terms of semantics relatedness by creating tolerance classes of terms in $\mathcal{T}$ and approximations of subsets of documents. Terms are viewed better using overlapping classes, which can be generated by *tolerance relations* R (with reflexive and symmetric properties) in an universe U instead of an equivalence relation (with reflexive, symmetric and transitive properties) used in the original rough set model. For formulating tolerance classes of index terms of documents, we employ the co-occurrence of index terms in all documents from $\mathcal{D}$. Denote by $f_{d_j}(t_i)$ the number of occurrences of term t_i in d_j, and by $f_{\mathcal{D}}(t_i)$ the number of documents in $\mathcal{D}$ that term t_i occurs in, and by $f_{\mathcal{D}}(t_i, t_j)$ the number of documents in $\mathcal{D}$ in which two index terms t_i and t_j co-occur. We define an uncertainty function I depending on a threshold θ as

$$I_\theta(t_i) = \{t_j \mid f_{\mathcal{D}}(t_i, t_j) \geq \theta\} \cup \{t_i\} \tag{1}$$

It is clear that the function I_θ defined above satisfies the condition of $t_i \in I_\theta(t_i)$ and $t_j \in I_\theta(t_i)$ iff $t_i \in I_\theta(t_j)$ for any $t_i, t_j \in \mathcal{T}$, and so I_θ is both reflexive and symmetric. This function corresponds to a tolerance relation $\mathcal{I} \subseteq \mathcal{T} \times \mathcal{T}$ that $t_i \mathcal{I} t_j$ iff $t_j \in I_\theta(t_i)$, and $I_\theta(t_i)$ is the tolerance class of index term t_i. A vague inclusion function ν, which determines how much X is included in Y, is defined as $\nu(X, Y) = |X \cap Y|/|X|$. Using this the membership function μ for $t_i \in \mathcal{T}, X \subseteq \mathcal{T}$ can be defined as $\mu(t_i, X) = \nu(I_\theta(t_i), X) = |I_\theta(t_i) \cap X|/|I_\theta(t_i)|$. With these definitions we can define a tolerance space as $\mathcal{R} = (\mathcal{T}, I, \nu, P)$ in which the *lower approximation* $\mathcal{L}(\mathcal{R}, X)$ and the *upper approximation* $\mathcal{U}(\mathcal{R}, X)$ in $\mathcal{R}$ of any subset $X \subseteq \mathcal{T}$ can be defined as

$$\mathcal{L}(\mathcal{R}, X) = \{t_i \in \mathcal{T} \mid \nu(I_\theta(t_i), X) = 1\} \tag{2}$$

$$\mathcal{U}(\mathcal{R}, X) = \{t_i \in \mathcal{T} \mid \nu(I_\theta(t_i), X) > 0\} \tag{3}$$

Index terms in each document are weighted as follows.

$$w_{ij} = \begin{cases} (1 + \log(f_{d_j}(t_i))) \times \log \frac{M}{f_{\mathcal{D}}(t_i)} & \text{if } t_i \in d_j, \\ \min_{t_h \in d_j} w_{hj} \times \frac{\log(M/f_{\mathcal{D}}(t_i))}{1+\log(M/f_{\mathcal{D}}(t_i))} & \text{if } t_i \in \mathcal{U}(\mathcal{R}, d_j) \setminus d_j \\ 0 & \text{if } t_i \notin \mathcal{U}(\mathcal{R}, d_j) \end{cases} \tag{4}$$

The vector length normalization is then applied to document representation.

3 TRSM-Based Hierarchical Clustering Algorithm

Figure 1 describes the general TRSM-based hierarchical clustering algorithm that is an extension of the hierarchical agglomerative clustering algorithm. The main point here is at each merging step where upper approximations of documents are used in finding two closest clusters to merge by group-average link. It allows to use cluster representatives to calculate the similarity between clusters instead of averaging similarities of all document pairs included in clusters with average complexity $O(N^2)$. This algorithm constructs a *polythetic* representative R_k for each cluster $C_k, k = 1, \ldots, K$. The following rules form the cluster representatives: (i) Initially, $R_k = \phi$, (ii) For all $d_j \in C_k$ and for all $t_i \in d_j$, if $f_{C_k}(t_i)/|C_k| > \sigma$ then $R_k = R_k \cup \{t_i\}$, (iii) If $d_j \in C_k$ and $d_j \cap R_k = \phi$ then $R_k = R_k \cup \text{argmax}_{t_i \in d_j} w_{ij}$. The weights of terms t_i in R_k are first averaged by of weights of this terms in all documents belonging to C_k, that means

$w_{ik} = (\sum_{d_j \in C_k} w_{ij})/|\{d_j : t_i \in d_j\}|$, then normalized by the length of the representative R_k. Three common coefficients of Dice, Jaccard and Cosine [1] are implemented to calculate the similarity between pairs of documents d_{j_1} and d_{j_2}. Two main advantages of using upper approximations are: (i) To reduce the

```
1.   Given: a collection of $M$ documents $\mathcal{D} = \{d_1, d_2, \ldots, d_M\}$
2.          a similarity measure $sim : \mathcal{P}(\mathcal{D}) \times \mathcal{P}(\mathcal{D}) \to R$
3.   **for** $j = 1$ **to** $M$ **do**
4.   $C_j = \{d_j\}$ **end**
5.   $H = \{C_1, C_2, , \ldots, C_M\}$
6.   $i = M + 1$
7.   **while** $|H| > 1$
8.          $(C_{n_1}, C_{n_2}) = \text{argmax}_{(C_u,C_v) \in H \times H} sim(\mathcal{U}(\mathcal{R}, C_u), \mathcal{U}(\mathcal{R}, C_v)$
9.          $C_i = C_{n_1} \cup C_{n_2}$
10.         $H = (H \setminus \{C_{n_1}, C_{n_2}\}) \cup \{C_i\}$
11.         $i = i + 1$
```

Fig. 1. TRSM-based hierarchical agglomerative clustering algorithm

number of zero-valued coefficients, and (ii) The upper approximations formed by tolerance classes make it possible to relate documents that may have few (even no) terms in common with the user's topic of interest or the query.

4 Validation and Evaluation

The first columns of Table 1 summarizes test collections used in our experiments. The clustering quality for each test collection depends on parameter θ in TRSM and on σ in clustering algorithm. Note that the higher value of θ the large upper approximation and the smaller lower approximation of a set X. In Table 3 we can see how retrieval effectiveness relates to different values of θ. To avoid biased experiments when comparing algorithms we take default values $\theta = 15$,

Table 1. Results of clustering tendency

		nb	nb	nb	nb rel.	% average of relevant doc.						
Col	Subject	doc	terms	queries	doc.	0	1	2	3	4	5	average
JSAI	Art. Int.	802	1813	20	32	19.9	19.8	18.5	18.5	11.8	11.5	2.2
CACM	Comp. Sci.	3200	6520	64	15	50.3	22.5	12.8	7.9	4.2	2.3	1.0
CISI	Library	1460	4414	76	40	45.4	25.8	15.0	7.5	4.3	1.9	1.1
CRAN	Aeronautics	1400	3182	225	8	33.4	32.7	19.2	9.0	4.6	1.0	1.2
MED	Medicine	1033	4841	30	23	10.4	18.7	18.6	21.6	19.6	11.1	2.5

Table 2. Synthesized results about the stability

	Percentage of changed data						
	1%	2%	3%	4%	5%	10%	15%
$\theta = 2$	2.84	5.62	7.20	5.66	5.48	11.26	14.41
$\theta = 3$	3.55	4.64	4.51	6.33	7.93	12.06	15.85
$\theta = 4$	0.97	2.65	2.74	4.22	5.62	8.02	13.78

and when comparing algorithms, default values $\theta = 15$, and $\sigma = 0.1$ for all five test collections.

4.1 Validation of Clustering Tendency

We employ the *nearest neighbor test* by considering, for each relevant document of a query, how many of its n nearest neighbors are also relevant; and by averaging over all relevant documents for all queries in a test collection in order to obtain single indicators. Table 1 reports the experimental results synthesized from those done on five test collections. Columns from 7 to 12 stand for the percentage average of the relevant documents in a collection that had 0, 1, 2, 3, 4, and 5 relevant documents in their sets of 5 nearest neighbors. The last column shows the average number of relevant documents among 5 nearest neighbors of each relevant document. The result on tendency suggests that the TRSM clustering method is appropriate for the retrieval purpose.

4.2 Validation of Clustering Stability

Table 2 shows the experimental results of clustering stability for JSAI test collection with different values of s from 210 experiments with $s\% = 1\%, 2\%, 3\%, 4\%, 5\%, 10\%$ and 15%. For each value 2, 3, and 4 of θ, the experiments are done 10 times each for a reduced database of size $(100-s)\%$ of $\mathcal{D}$. We randomly removed a specified amount of $s\%$ documents from the collection, then re-determined the new tolerance space for the reduced database to perform the TRSM clustering algorithm and evaluate the change of clusters due to the change of the database. Note that a little change of data implies a possible little change of hierarchy (about at the same percentage as for $\theta = 4$). The experiments for other test collections have nearly the same results. It suggests that the TRSM hierarchical clustering are stable.

4.3 Evaluation of Cluster-Based Retrieval Effectiveness

The experiments evaluate effectiveness of the TRSM cluster-based retrieval by comparing it with full retrieval by using the common measures of *precision* and *recall*. Table 3 shows precision and recall of the TRSM-based full retrieval and the VSM-based full retrieval (Vector Space Model) where the TRSM-based retrieval is done with values 30, 25, 20, 15, 10, 8, 6, 4, and 2 of θ. We see that

Table 3. Precision and recall of full retrieval

	JSAI		CACM		CISI		CRAN		MED	
θ	P	R	P	R	P	R	P	R	P	R
30	0.934	0.560	0.146	0.231	0.147	0.192	0.265	0.306	0.416	0.426
25	0.934	0.560	0.158	0.242	0.151	0.194	0.266	0.310	0.416	0.426
20	0.934	0.560	0.159	0.243	0.150	0.194	0.268	0.311	0.416	0.426
15	**0.934**	**0.560**	**0.160**	**0.241**	**0.155**	**0.204**	**0.257**	**0.301**	**0.415**	**0.421**
10	0.934	0.560	0.141	0.221	0.142	0.178	0.255	0.302	0.414	0.387
8	0.934	0.560	0.151	0.254	0.138	0.172	0.242	0.291	0.393	0.386
6	0.945	0.550	0.141	0.223	0.146	0.178	0.233	0.271	0.376	0.365
4	0.904	0.509	0.137	0.182	0.152	0.145	0.223	0.241	0.356	0.383
2	0.803	0.522	0.111	0.097	0.125	0.057	0.247	0.210	0.360	0.193
VSM	**0.934**	**0.560**	**0.147**	**0.232**	**0.139**	**0.184**	**0.258**	**0.295**	**0.429**	**0.444**

Table 4. Precision and recall of the TRSM cluster-based and full retrieval

	1.2% (0.18)		1.8% (0.16)		2.9% (0.14)		8.0% (0.11)		16.9% (0.09)		full search	
Col.	P	R	P	R	P	R	P	R	P	R	P	R
JSAI	0.950	0.472	0.948	0.485	0.949	0.502	0.939	0.541	0.938	0.559	0.934	0.560
CACM	0.048	0.037	0.096	0.068	0.100	0.084	0.116	0.194	0.105	0.262	0.160	0.241
CISI	0.181	0.043	0.180	0.061	0.180	0.089	0.130	0.183	0.112	0.261	0.155	0.204
CRAN	0.121	0.127	0.140	0.149	0.139	0.173	0.139	0.214	0.112	0.245	0.257	0.301
MED	0.477	0.288	0.530	0.324	0.565	0.375	0.518	0.460	0.422	0.531	0.415	0.421

Table 5. Precision and recall of the TRSM and VSM cluster-based retrieval

	2.9% of $\mathcal{D}$ ($\gamma = 0.14$)				8.0% of $\mathcal{D}$ ($\gamma = 0.11$)				16.9% of $\mathcal{D}$ ($\gamma = 0.09$)			
	TRSM		VSM		TRSM		VSM		TRSM		VSM	
Col.	P	R	P	R	P	R	P	R	P	R	P	R
JSAI	0.949	0.502	0.947	0.501	0.939	0.541	0.947	0.518	0.938	0.559	0.939	0.549
CACM	0.100	0.084	0.075	0.479	0.116	0.194	0.075	0.479	0.105	0.262	0.075	0.479
CISI	0.180	0.089	0.099	0.366	0.130	0.183	0.099	0.366	0.112	0.261	0.099	0.366
CRAN	0.139	0.173	0.066	0.519	0.139	0.214	0.066	0.519	0.112	0.245	0.066	0.519
MED	0.565	0.375	0.520	0.430	0.518	0.460	0.458	0.521	0.422	0.531	0.375	0.585

precision and recall for JSAI are high, and they are higher and stable for the other collections with $\theta \geq 15$. With these values of θ, the TRSM-based retrieval effectiveness is comparable or somehow higher than that of VSM. Table 4 reports the average of precision and recall for all queries in experiments for TRSM cluster-based retrieval in a subset $\mathcal{D}'$ of $\mathcal{D}$ that contains clusters closed to the query. We experiment with various proportion (%) of $|\mathcal{D}'|$ to $|\mathcal{D}|$, and full retrieval in whole $\mathcal{D}$ (accordingly, values of γ). In several cases (JSAI, CISI, and MED) just searching a small part of $\mathcal{D}$, says 1.2% or 1.8%, TRSM cluster-based search reaches precision higher than that of full search. Also, the TRSM cluster-based search achieved recall higher than that of full retrieval on most collections when $|\mathcal{D}'|$ is about 17% of $|\mathcal{D}|$. Table 5 reports the effectiveness of TRSM cluster-based

Table 6. Performance Measurements of the TRSM Cluster-based Retrieval

Col.	Size (MB)	Nb of Queries	TRSM Time	Clustering Time	Full Search (sec)	Cluster Search (sec)	Memory (MB)
JSAI	0.1	20	2.4s	14.9s	0.8	0.1	8
CACM	2.2	64	22m2.2s	26m46.8s	13.3	1.2	201
CISI	2.2	76	13m16.8s	4m49.8s	40.1	3.4	84
CRAN	1.6	225	23m9.9s	3m6.9s	20.5	1.8	71
MED	1.1	30	40.1s	1m30.8s	2.5	0.3	25

retrieval (TRSM) versus VSM cluster-based retrieval (VSM) when $|\mathcal{D}'|$ is 2.9%, 8.0%, and 16.9% of $|\mathcal{D}|$. It shows that TRSM cluster-based retrieval often achieves precision higher than that of VSM cluster-based retrieval thought its recall is somehow lower.

4.4 Evaluation of TRSM Hierarchical Clustering Efficiency

A direct implementation of procedures in the first phase requires the time complexity of $O(M + N^2)$, but we implemented them by applying the quick-sort algorithm of $O(N \log N)$ to make the indexing files, then created the TRSM related files for the term co-occurrence, tolerance classes, upper and lower approximations files in the time of $O(M + N)$. We can note that the search for clusters requires in average $\log M$, then the search will be done with a subset of documents in the clusters. However, the time complexity of the clustering is of $O(M^2 + N)$, and the space is of $O(M^2 + N)$ because of using an $M \times M$-matrix to store the similarities/distances between clusters in the hierarchy. All the experiments reported in this paper were performed on a conventional workstation GP7000S Model 45 (Fujitsu, 250 MHz Ultra SPARC-II, 512 MB). Table 6 summaries the time for generating the TRSM files, clustering, full search, cluster-based search, and the required memory size for each collection.

References

1. Fakes, W. B. and Baeza-Yates, *Information Retrieval. Data Structures and Algorithms* (eds.), Prentice Hall, 1992.
2. Ho, T. B. and Funakoshi K., “Information retrieval using rough sets”, *Journal of Japanese Society for Artificial Intelligence*, Vol. 13, N. 3, 1998, 424–433.
3. Pawlak, Z., *Rough sets: Theoretical Aspects of Reasoning about Data*, Kluwer Academic Publishers, 1991.
4. Polkowski, L. and Skowron, A., *Rough Sets in Knowledge Discovery 2. Applications, Case Studies and Software Systems* (eds.), Physica-Verlag, 1998.
5. Skowron, A. and Stepaniuk, J., “Generalized approximation spaces”, *The 3rd International Workshop on Rough Sets and Soft Computing*, 1994, 156–163.