Tolerance Rough Set Model Approach to Document Clustering

Tu Bao Ho, Ngoc Binh Nguyen and Saori Kawasaki
School of Knowledge Science
Japan Advanced Institute of Science and Technology
Tatsunokuchi, Ishikawa, 923-1292 JAPAN

1 A tolerance rough set model

Denote the set of M full text documents by $\mathcal{D}$, and the set of N terms from $\mathcal{D}$ by $\mathcal{T}$. The method generates a hierarchical structure of $\mathcal{D}$ in two phases. The first phase extracts and maps each document $d_j \in \mathcal{D}$ into a list of terms t_i each of which is assigned a weight that reflects its importance in the document, then enriches documents with their approximations by the proposed tolerance rough set model. The second phase groups documents by an agglomerative clustering method using the document approximations.

The key issue in formulating the *tolerance rough set model* (TRSM) to represent documents is the identification of tolerance classes of index terms. We employ the co-occurrence of index terms in all documents from $\mathcal{D}$ to determine a tolerance relation and tolerance classes. 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

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

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

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

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

Index terms in each document are weighted by usual ways in natural language processing [2] with modifications for the upper approximations. The vector length normalization is then applied to document representation.

2 TRSM-based Clustering

Table 1 presents the idea of TRSM-based hierarchical clustering algorithm that is an extension of the hierarchical agglomerative clustering algorithm. The main point is at each merging step it uses document upper approximations in finding two closest clusters to merge.

1. Initially, consider each document of $\mathcal{D}$ as a cluster with one member $C_j = \{d_j\}$, and $H = \{C_1, C_2, \ldots, C_M\}$.
2. Find two most similar clusters in terms of upper approximations of their representatives, $(C_{n_1}, C_{n_2}) = \text{argmax}_{(C_u, C_v) \in H \times H} \mathcal{S}(\mathcal{U}(\mathcal{R}, C_u), \mathcal{U}(\mathcal{R}, C_v))$
3. Form a new cluster $C_i = C_{n_1} \cup C_{n_2}$ and let $H = (H \setminus \{C_{n_1}, C_{n_2}\}) \cup \{C_i\}$.
4. If more than one cluster remains, return to 2 and 3.

Table 1: TRSM-based hierarchical clustering

Table 2 presents the idea of TRSM nonhierarchical clustering algorithm. It is an extension of the reallocation clustering method with tolerance classes in forming K clusters of a collection $\mathcal{D}$. Two issues have been carefully considered: polythetic representation of clusters and distance between documents.

1. Determine the initial representatives $R_1, R_2, ..., R_K$ of clusters $C_1, C_2, ..., C_K$ as K randomly selected documents in $\mathcal{D}$.
2. For $d_j \in \mathcal{D}$, calculate similarity $S(\mathcal{U}(\mathcal{R}, d_j), R_k)$ and the cluster representative $R_k, k = 1, ..., K$. If the similarity is greater than a threshold, assign d_j to C_k and take it as the membership value $m(d_j)$ of d_j in C_k.
3. For each C_k, re-determine its representative R_k.
4. Repeat steps 2 and 3 until there is little or no change in cluster membership during a pass through $\mathcal{D}$.
5. Assign unclassified documents to formed clusters using their nearest neighbors.

Table 2: TRSM nonhierarchical clustering

3 Validation and Evaluation

We do validation on clustering tendency and stability, and evaluation of cluster-based effectiveness and efficiency in application to information retrieval. Experiments are done with the Japanese Society for AI

Col.	1.2% (0.18)		1.8% (0.16)		2.9% (0.14)		8.0% (0.11)		16.9% (0.09)		full search	
	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

Col.	1 cluster		2 clusters		3 clusters		4 clusters		5 clusters		full search	
	P	R	P	R	P	R	P	R	P	R	P	R
JSAI	0.973	0.375	0.950	0.458	0.937	0.519	0.936	0.544	0.932	0.534	0.934	0.560
CACM	0.098	0.063	0.100	0.127	0.117	0.166	0.132	0.221	0.144	0.240	0.160	0.241
CISI	0.177	0.078	0.141	0.139	0.151	0.179	0.156	0.206	0.158	0.212	0.155	0.204
CRAN	0.204	0.219	0.238	0.278	0.250	0.290	0.257	0.301	0.261	0.304	0.257	0.301
MED	0.393	0.277	0.396	0.393	0.372	0.425	0.367	0.445	0.380	0.472	0.415	0.421

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

database (JSAI) and four test collections [1]. We take values $\theta = 15$, $\sigma = 0.1$ for all test collections.

3.1 Validation of Clustering

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 3 reports the experimental results synthesized from those done on five test collections. The result on tendency suggests that the TRSM clustering method is appropriate for the retrieval purpose.

	% average of relevant documents						av
	0	1	2	3	4	5	
JSAI	19.9	19.8	18.5	18.5	11.8	11.5	2.2
CACM	50.3	22.5	12.8	7.9	4.2	2.3	1.0
CISI	45.4	25.8	15.0	7.5	4.3	1.9	1.1
CRAN	33.4	32.7	19.2	9.0	4.6	1.0	1.2
MED	10.4	18.7	18.6	21.6	19.6	11.1	2.5

Table 3: Results of clustering tendency

Table 4 shows results of clustering stability for JSAI 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 JSAI database, then re-determine the new tolerance space for the reduced database. The experiments on the stability for other test collections have nearly the same results as those of JSAI, and these suggest that the TRSM clustering results are stable.

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

Table 4: Synthesized results about the stability

3.2 Evaluation of Clustering

We evaluate effectiveness of TRSM cluster-based retrieval by using the common measures of *precision* and *recall*. The upper part of Table 5 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. The lower part of Table 5 reports the average of precision and recall for all queries in test collections using the TRSM cluster-based retrieval with 1, 2, 3, 4 clusters, and full retrieval (15 clusters). It suggests that TRSM cluster-based retrieval often achieves precision higher than that of VSM cluster-based retrieval thought its recall is somehow lower.

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.

References

[1] Fakes, W. B. and Baeza-Yates, *Information Retrieval. Data Structures and Algorithms* (eds.), Prentice Hall, 1992.

[2] Manning, C. D. and Schutze, H., *Foundations of Statistical Natural Language Processing*, The MIT Press, 1999.