

# A Patent Retrieval Method Using a Hierarchy of Clusters at TUT

Hironori Doi Yohei Seki Masaki Aono

Toyohashi University of Technology

1-1 Hibarigaoka, Tenpaku-cho, Toyohashi-shi, Aichi 441-8580, Japan

doi@kde.ics.tut.ac.jp, {seki,aono}@ics.tut.ac.jp

## Abstract

*To retrieve relevant documents from an enormous document collection, we usually utilize the similarity or distance measure between a query and the documents, or apply document clustering techniques to the document collection and partition it into relevant document groups. For patent retrieval, however, it is difficult to retrieve documents by using query terms only, because complex terminologies specific to patents appear in them. One approach to solving this problem is to use query expansion techniques. We have extended the usual vector space model by utilizing co-clustering techniques. We generate a hierarchy of clusters by applying these techniques to the document collection with different levels of cluster granularity. The query is then expanded by using this hierarchy of clusters. We participated in the NTCIR-5 Patent Retrieval Task (Document Retrieval Subtask) using our system and present the effectiveness of our approach for patent retrieval with experiments using the NTCIR-4 and NTCIR-5 test collections.*

**Keywords:** *Query Expansion, Document Clustering.*

## 1 Introduction

Patent data is usually classified into different groups with codes such as IPC (International Patent Classification), which classify themes from a single viewpoint using names, summaries, or claims, or by using F-terms, which classify themes from multiple viewpoints using narratives. On the other hand, document clustering techniques, which partition the document set into groups according to the similarity or distance measure between the elements in the document set, are widely used and applied to retrieving documents.

For patent data, concept-level searches based on the similarity of contents are more effective than exact keyword matching using queries, because different terms are frequently used to express the same concept.

In this paper, we explore our approach to the NTCIR-5 Patent Retrieval Task (Document Retrieval

Subtask). We have previously proposed the approach in [1] as a query expansion method with a hierarchy of clusters generated by applying co-clustering techniques to the document collection at different levels of cluster granularity. Cluster granularity was defined using powers of two (16, 32, 64, ...).

## 2 Related Work

Beginning in the 1990s, researchers have applied clustering techniques to information retrieval. Cutting et al. [3] applied an exclusive clustering technique to retrieval results using the  $k$ -means algorithm. Eguchi et al. [6] proposed a retrieval method using clustering similar to Cutting's and relevance feedback techniques, and they reported improvements in search effectiveness. Sasaki et al. [7] improved the effectiveness of retrieval by using spherical  $k$ -means clustering and reducing dimensions with a random projection technique. Chang et al. [2] proposed an automated query expansion method using clustering features.

## 3 Co-clustering

Recent work, which used clustering to improve the search effectiveness, was restricted only to document clustering. We propose a retrieval method using not only document clusters but term clusters by utilizing co-clustering [5, 4] techniques to improve search effectiveness, especially for a concept-level search. This proposal is based on the motivation that a set of co-occurring terms in the document is an effective index to retrieve relevant special-use documents such as patents. In this section, we give an overview of co-clustering. We present our proposal in the next section.

### 3.1 Definition

Let  $X$  and  $Y$  be two discrete random variables that take values in a document set  $X = \{x_1, x_2, \dots\}$  and a term (index) set  $Y = \{y_1, y_2, \dots\}$  respectively.

Suppose that we know their joint probability distribution  $p(X, Y)$ ; often this can be estimated using co-occurrence data, expressed as a (normalized) matrix consisting of co-occurrences of terms and documents. Let  $\hat{X} = \{h_1, h_2, \dots, h_s\}$  be  $s$  document clusters, and  $\hat{Y} = \{g_1, g_2, \dots, g_t\}$  be  $t$  term clusters. Let  $\pi_x \equiv p(x)$  denote the probability density function that a document  $x$  appears, and  $\pi_y \equiv p(y)$  denote the probability density function that a term  $y$  appears. For clusters, let  $\pi_h \equiv p(x) = \sum_{x \in h} p(x)$  be the probability density function of a document cluster  $h$ , and let  $\pi_g \equiv p(g) = \sum_{y \in g} p(y)$  be the probability density function of a term cluster  $g$ .

Given a document  $x$ , the probability density function that a set of terms  $Y$  appears in  $x$  can be expressed as  $z(x) = p(Y | x)$ . Similarly, given a document cluster  $h$ , the probability density function that a set of terms  $Y$  appears in  $h$  can be expressed as  $\mu_h(x) = p(Y | h)$ . In the co-clustering algorithm, an optimal document cluster  $\hat{X}$  and an optimal term cluster  $\hat{Y}$  are found subject to the condition that minimizes the mutual information  $I(X, Y) - I(\hat{X}, \hat{Y})$ , considering the co-occurring terms in the documents.

$$I(X, Y) - I(\hat{X}, \hat{Y}) = KL(p(X, Y) \| q(X, Y)) \quad (1)$$

$KL(\cdot \| \cdot)$  on the right-hand side of the above expression stands for the Kullback–Leibler divergence. We assume the following conditions:  $q(x, y) = p(h, g)p(x | h)p(y | g)$  and  $p(h, g) = \sum_{x \in h} \sum_{y \in g} p(x, y)$ .

### 3.2 Advantages of Co-clustering

With the co-clustering algorithm, document clusters and term clusters are generated together by minimizing the difference in mutual information between the original random variables and the clustered random variables. This implies that by using co-clustering, a document–term matrix with low dimensions that approximates the original matrix preserves highly associated local subspaces. This is in contrast to singular value decomposition (SVD), in that SVD usually reduces dimension by a global effect on the matrix and generates a dense matrix in which the elements consist of both positive and negative values. In co-clustering, the effect of approximation is localized and preserves the sparseness of the original matrix, and it does not contain negative values. This is preferable, because there are no negative values in document–term matrices that contain TF-IDF values.

## 4 Our Proposed Method

We have developed a hierarchical data structure “granularity levels of clusters” to cope with the problems inherent in most clustering methods discussed in

4.1. In this section, we will briefly enumerate problems of typical clustering, then describe the “cluster averaging algorithm”, the “hierarchical cluster generation algorithm”, and a query expansion method based on these algorithms.

### 4.1 Clustering Problems

We have three problems with clustering:

1. The quality of the clusters depends heavily on the initial parameters of the clustering algorithms.
2. Partition-based clustering suffers from noisy data.
3. The number of clusters to be generated cannot be properly determined a priori.

We propose three solutions to these problems. For the dependency on initial parameters, we change the initial seed for the random variables, run the clustering algorithms more than once, and average the clusters into similar clusters. This algorithm is described as the “Cluster Averaging Algorithm” in Section 4.1.1.

For the noise problem, we compute the similarity between documents in each cluster, and remove noisy documents that were not contained in any clusters in the cluster averaging algorithm. We therefore generate an effective partitioning into clusters.

For the problem of the number of clusters, we generate clusters at many granularity levels. We then link similar clusters between different granularity levels of clusters (16, 32, 64, ...) above the threshold, and make a hierarchical cluster data structure. This structure is shown in Figure 1. This algorithm is described as the “Hierarchical Cluster Generation Algorithm” in Section 4.1.2.

#### 4.1.1 Cluster Averaging Algorithm

Step 1 Perform clustering with granularity level  $M$  ( $M_1, \dots, M_k$ )  $R$  times by changing the initial seed for the random variables. Denote the resulting document clusters by  $\mathbb{D}^r = \{D_1^r, D_2^r, \dots, D_M^r\}$  ( $r = 1, \dots, R$ ) and the term clusters by  $\mathbb{W}^r = \{W_1^r, W_2^r, \dots, W_M^r\}$  ( $r = 1, \dots, R$ ). The document cluster  $D_j^r$  corresponds to the term cluster  $W_j^r$ .

Step 2 Initialize a vector  $\mathbb{H} = \{H_1, H_2, \dots, H_M\}$  (which we call the “document cluster vector”) by  $\mathbb{D}^1$ . In the same way, initialize a vector  $\mathbb{G} = \{G_1, G_2, \dots, G_M\}$  (which we call the “term cluster vector”) by  $\mathbb{W}^1$ .

Step 3 Apply the cluster smoothing algorithm shown in Figure 2. In this step, we perturb the cluster vector of  $\mathbb{H}$  if the currently scanned (processed) document vector  $\mathbb{D}$  has similarity to  $\mathbb{H}$  above a pre-defined threshold  $\delta$ . We call this step “cluster smoothing”.

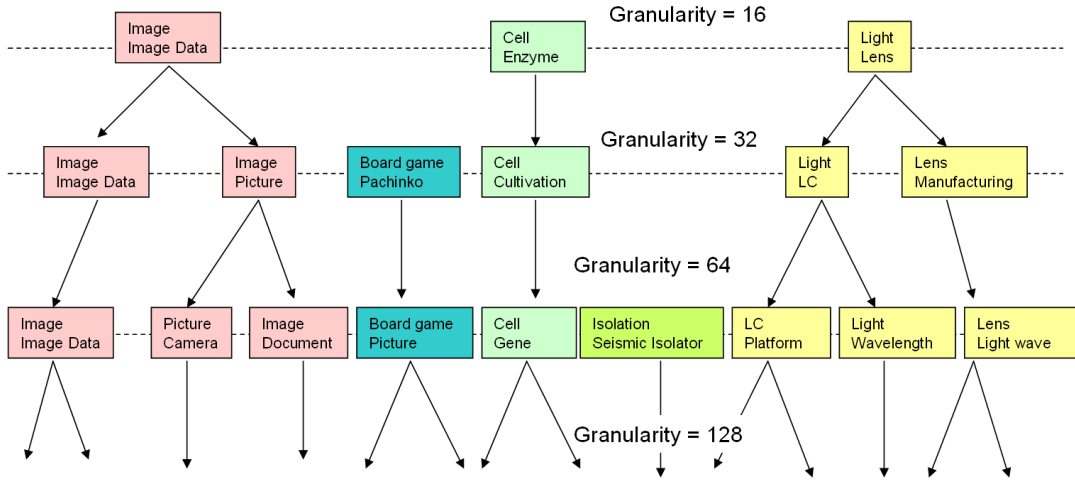


Figure 1. A hierarchical cluster data structure

#### 4.1.2 Hierarchical Cluster Generation Algorithm

- Step 1 Compute the similarity between  $\mathbb{H}_i$  and  $\mathbb{H}_{i+1}$  using the average cluster vectors.
- Step 2 If the similarity between the document cluster  $D_i$  in  $\mathbb{H}_i$  and the document cluster  $D_{i+1}$  in  $\mathbb{H}_{i+1}$  is larger than a threshold, add a link from  $D_i$  to  $D_{i+1}$ .
- Step 3 Repeat [Step 2] until the granularity level reaches the finest document cluster set  $\mathbb{H}_k$ .

### 5 Query Expansion

In this section, we explain our query expansion method using a generated hierarchy of clusters.

With a hierarchical cluster granularity data structure, we compute the similarity between a query vector and the average cluster vectors in order of the roughness of granularity. If the similarity is above a threshold value, queries are expanded by adding the terms and the weights in the average cluster vectors, as in the following expression:

$$\mathbf{q}' = \alpha \mathbf{q} + \sum w_i \mathbf{v}_i \quad (2)$$

Note that  $v_i$  represents an expanded word vector,  $w_i$

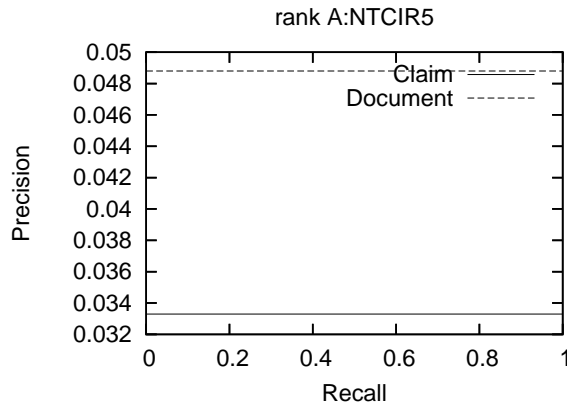
#### Algorithm ClusterSmoothing( $\mathbb{H}, \mathcal{G}, R, M$ )

```

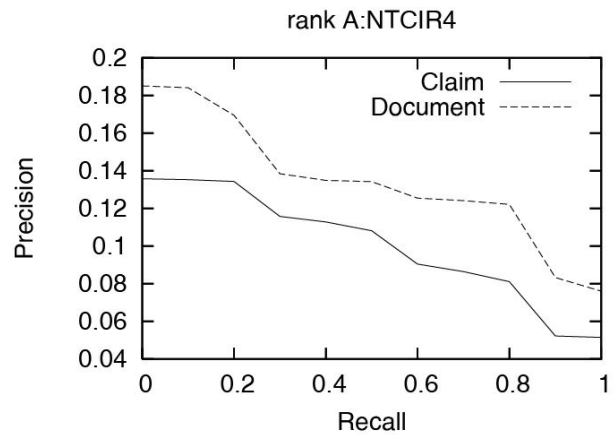
1: for  $r = 2$  to  $R$ 
2:   for  $j = 1$  to  $M$ 
3:      $\mathbf{D}_i^r = \{\mathbf{d}_{i,1}^r, \mathbf{d}_{i,2}^r, \dots, \mathbf{d}_{i,k_0}^r\}$ ;
4:      $\mathbf{W}_i^r = \{w_{i,1}^r, w_{i,2}^r, \dots, w_{i,l_0}^r\}$ ;
5:     for  $i = 1$  to  $|\mathbb{H}|$ 
6:        $\mathbf{H} = \mathbf{H}_i$ ;  $\mathbf{D} = \mathbf{D}_i^r$ ;  $\mathbf{W} = \mathbf{W}_i^r$ ;
7:       if (Similarity( $\mathbf{H}, \mathbf{D}$ ) >  $\delta$ ) then
8:          $\mathbf{H}_i = \text{Average}(\mathbf{H}, \mathbf{D}, \mathbf{W})$ ;
9:       else
10:         $\mathbb{H} = \mathbb{H} \cup \mathbf{D}$ ;
11:         $\mathcal{G} = \mathcal{G} \cup \mathbf{W}$ ;
12:      endif
13:    end for
14:  end for
15: end for

```

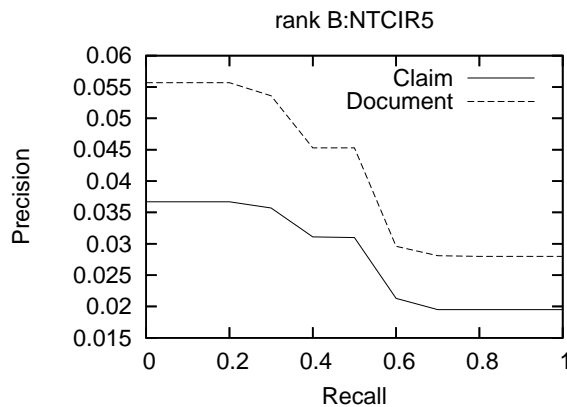
Figure 2. Cluster smoothing algorithm



**Figure 3. Evaluation A on NTCIR-5 Patent Retrieval Task (Document Retrieval Sub-task)**



**Figure 5. Evaluation A on NTCIR-4 Patent Test Collection**



**Figure 4. Evaluation B on NTCIR-5 Patent Retrieval Task (Document Retrieval Sub-task)**

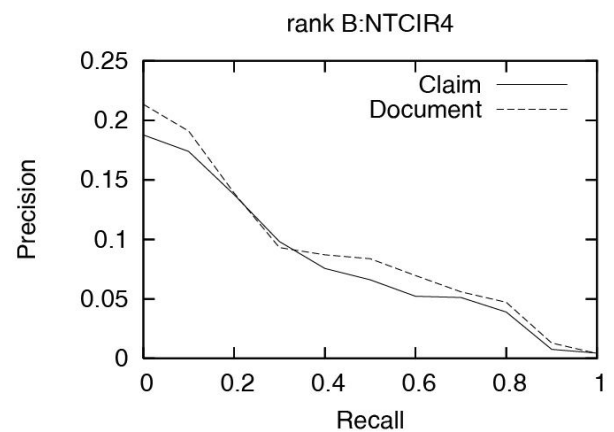
represents its weight, and  $\alpha$  is a nonnegative coefficient.

Clusters found at coarser granularity levels are basically major clusters, and the number of elements is generally large. This is preferable for applications that focus on recall and that search similar data as much as possible.

## 6 Results

We participated in the NTCIR-5 Patent Retrieval Task (Document Retrieval Subtask) using our system. The results are shown in Figures 3 and 4. We also conducted additional experiments using the NTCIR-4 Patent Test Collection. The results are shown in Figures 5 and 6.

In the figures, “claim” refers to the result with query expansion using only terms in patent claims as queries,



**Figure 6. Evaluation B on NTCIR-4 Patent Test Collection**

