Novelty-based Incremental Document Clustering for On-line Documents

Sophoin Khy[1] Yoshiharu Ishikawa[2,3] Hiroyuki Kitagawa[2,3]
[1]*Master's Program in Science and Engineering*
[2]*Graduate School of Systems and Information Engineering*
[3]*Center for Computational Sciences*
University of Tsukuba
Tsukuba, Ibaraki, Japan
sophoin@kde.cs.tsukuba.ac.jp, {ishikawa,kitagawa}@cs.tsukuba.ac.jp

Abstract

Document clustering has been used as a core technique in managing vast amount of data and providing needed information. In on-line environments, generally new information gains more interests than old one. Traditional clustering focuses on grouping similar documents into clusters by treating each document with equal weight. We proposed a novelty-based incremental clustering method for on-line documents that has biases on recent documents. In the clustering method, the notion of 'novelty' is incorporated into a similarity function and a clustering method, a variant of the K-means method, is proposed. We examine the efficiency and behaviors of the method by experiments.

1. Introduction

In the recent years, the proliferation of on-line information services that distribute news, emails, weblogs, etc., has led to the increase of vast amount of on-line information sources on the Internet. Given myriad electronic documents, it is difficult for users to find needed information.

Document clustering [2, 3, 9] is a method which groups documents into clusters such that documents in the same cluster are similar to each other, whereas documents in different clusters are dissimilar. It has been used as a fundamental method in many areas such as data mining [5], information retrieval [4], topic detection and tracking [1], and as a preprocessing step for other algorithms such as text classification and news summarization [10, 11].

Generally, in on-line environments, users tend to be interested in new and up-to-date information, especially in the case of news articles. Thus we proposed a novelty-based document clustering in [8]. *Novelty-based document clustering* is a document clustering technique that puts high weights on recent documents and low weights on old ones. The goal of the method is to produce clustering results reflecting current trends of hot topics.

In this paper, we present a novelty-based incremental document clustering method by proposing a new algorithm based on the *K*-means method. We will show two experiments. The objective of the first experiment is to evaluate the efficiency of our clustering technique by comparing the computation time of incremental version with non-incremental version of our method. We will show the efficiency of the incremental version of our proposed method. The objective of the second experiment is to examine whether the question: "what are recent topics?" is answered. We will show the results of experiments on different values of parameters and their evaluation.

The remainder of this paper is organized as follows. We start in Section 2 by reviewing related work on Topic Detection and Tracking and clustering. Section 3 describes the similarity measure. In Section 4 we introduce the *K*-means method and our proposed extended *K*-means method. Section 5 gives incremental statistics update and clustering. Section 6 presents experiments and results. Section 7 concludes the paper and discusses future work.

2. Related work

2.1. Topic detection and tracking

Topic detection and tracking (TDT) is a research project organized by NIST [1]. TDT addresses multiple sources of information, including both text and speech. These sources are news wires, radio and television news broadcast programs, and WWW sources. Several TDT evaluation competitions were held to evaluate research progress and technical capabilities. TDT research tries to analyze on-line documents such as broadcast news based on the notions of events and topics. The definition of TDT tasks changes every year. Generally, there are 5 research tasks defined in the TDT program [7]:

- Story segmentation - Detect changes between topically cohesive sections,
- Topic tracking - Keep track of stories similar to a set of example stories,
- Topic detection - Build clusters of stories that discuss the same topic,
- First story detection - Detect if a story is the first story of a new, unknown topic,

Proceedings of the 22nd International Conference on Data Engineering Workshops (ICDEW'06)
0-7695-2571-7/06 $20.00 © 2006 IEEE

- Link detection - Detect whether or not two stories are topically linked.

Clustering technique is widely used in many TDT tasks. However, the goals of TDT tasks are to break an arriving stream of text from newswire sources into individual news stories, to monitor the stories for events that have not been seen before, and to gather the stories into groups such that each group discusses a single news topic [1]. In the TDT detection task, various clustering techniques are used, but the goal of the task is to generate clusters of stories that discuss the same topic, while the goal of our novelty-based clustering method is to present an overview of the current trend of hot topics in the clustering results. Since the goal of our research is different from the TDT research, we can not use the TDT evaluation framework directly in our research.

2.2. Document clustering

Traditional document clustering can be categorized into two main categories: the hierarchical and partitioning methods. Hierarchical clustering methods cluster documents hierarchically whereas partitioning methods cluster documents by producing flat clusters.

Conventional hierarchical clustering methods are the single link method, the complete link method and the group average link method. Partitioning clustering methods include the *K*-means method, the *K*-medoids method, etc. The widely used partitioning method is the *K*-means method. The method is well-known for its efficiency compared to the hierarchical method.

Yang et al. proposed a method called GAC (group-average clustering) [12] extending Cutting's Fractionation method [3] by introducing temporal bucketing and reclustering. News stories are bucketed based on the order in which they are reported. That is, the method gives a higher priority to grouping temporally proximate stories than to temporally disparate ones. GAC divides chronologically ordered news stories into buckets and performs group average method to the buckets and repeatedly forms clusters hierarchically until a specified condition is met. GAC periodically reclusters the stories within each of the top level clusters by flattening the component clusters and regrowing clusters internally from the leaf nodes.

Yang et al. also proposed the single-pass incremental clustering (INCR) in [12]. INCR sequentially processes the input documents, one at a time, and grows clusters incrementally. A new document is assigned to a previous cluster if the similarity score between the document and the cluster is above a preselected threshold. Otherwise the document becomes the seed of a new cluster. The method introduces different thresholds for retrospective and on-line detection. INCR also imposes a time window in which the linear decaying-weight function is incorporated in the similarity function.

Compared with GAC, our proposed method also uses chronologically sorted input data and partitions the data into time windows. Each time window basically corresponds to one news program which includes multiple news articles. However, our approach does not process all the time windows in one clustering. When a new time window (a collection of news articles) arrives, a clustering process is triggered then the result is presented. In addition, GAC is based on a hierarchical method and uses the traditional cosine measure similarity function. Our novelty-based document clustering technique is based on a partitioning method, the *K*-means method, and incorporates the idea of novelty in the similarity function. Compared with INCR which also incorporates a time window and a linear decaying weight in the similarity function, our method uses exponential decaying factor by which weight of a document decreases exponentially. High weights are put on recent documents and low weights on old ones. Our objective is not only to generate clusters but also to reflect current trend of hot/recent topics.

F^2ICM (Forgetting-Factor-based Incremental Clustering Method) [8], the former version of our clustering method, devises a forgetting-factor-based similarity function and derives clustering method partially based on C^2ICM [2]. In the clustering algorithm, F^2ICM first computes the seeds from documents and then classifies documents sequentially based on the seeds. The difference between this paper and our previous paper [8] is that both papers share the same similarity formulas and incremental statistics update, but this paper is different from [8] mainly in the clustering criteria and algorithm, and more concrete experiments and evaluation.

3. Similarity measure

In this section, we briefly provide an overview of the novelty-based similarity function introduced in F^2ICM. The similarity measure of our clustering method is derived from the *document forgetting model* [8]. The model is based on a simple intuition: the values of on-line documents delivered everyday are considered to be gradually losing their values as time passes.

The model introduces the notion of a *forgetting factor*. Every document is assigned an initial weight *one* when it is acquired from its source. The document weight gradually decays as time passes according to the rate specified by the forgetting factor. The *weight* of document d_i at time τ is defined as:

$$dw_i \equiv \lambda^{\tau - T_i}, \tag{1}$$

in which λ ($0 < \lambda < 1$) is the *forgetting factor* and T_i ($T_i \leq \tau$) is the acquisition time of each document d_i. To set the parameter λ, we assume that the user gives a *half-life span* value β. It specifies the period that a document loses half of its weight. Namely, β satisfies $\lambda^\beta = 1/2$. Therefore, λ can be derived as

$$\lambda = \exp(-\log 2/\beta). \tag{2}$$

If n is the number of documents in the repository, the *total weight* of all documents is:

$$tdw \equiv \sum_{l=1}^{n} dw_l. \tag{3}$$

We define the subjective probability that the document d_i is randomly selected from the document set as:

$$\Pr(d_i) \equiv \frac{dw_i}{tdw}. \quad (4)$$

That is, when a document is acquired from a news provider, the selection probability $\Pr(d_i)$ of the document is $1/tdw$. As time passes, the selection probability decreases and approaches zero. This selection probability indicates the effect that our approach is 'forgetting' old documents.

The conditional probability that a document d_j is obtained when d_i is given is:

$$\begin{aligned} \Pr(d_j|d_i) &= \sum_{k=1}^{n} \Pr(d_j|d_i, t_k) \Pr(t_k|d_i) \\ &\simeq \sum_{k=1}^{n} \Pr(d_j|t_k) \Pr(t_k|d_i). \end{aligned} \quad (5)$$

The co-occurrence probability of document d_i and d_j is:

$$\begin{aligned} \Pr(d_i, d_j) &= \Pr(d_j|d_i) \cdot \Pr(d_i) \\ &\simeq \Pr(d_i) \sum_{k=1}^{m} \Pr(d_j|t_k) \Pr(t_k|d_i). \end{aligned} \quad (6)$$

This co-occurrence probability of the two documents is defined as the *similarity score* between them

$$sim(d_i, d_j) \equiv \Pr(d_i, d_j). \quad (7)$$

From the above similarity formula, we can say that the more a document d_i becomes old, the smaller its similarity scores with other documents as old documents have small $\Pr(d_i)$ values. By introducing the forgetting factor into similarity calculation, we can obtain clustering results based on similarity and novelty of documents.

The $\Pr(t_k|d_i)$ used in above formula is the occurrence probability of term t_k in document d_i

$$\Pr(t_k|d_i) \equiv \frac{f_{ik}}{\sum_{l=1}^{m} f_{il}}, \quad (8)$$

where f_{ik} is the number of occurrences of term t_k in document d_i. $\Pr(d_j|t_k)$ can be defined by the Bayes' theorem as

$$\Pr(d_j|t_k) = \frac{\Pr(t_k|d_j) \Pr(d_j)}{\Pr(t_k)}. \quad (9)$$

If *n* is the total number of documents, the occurrence probability of term t_k, $\Pr(t_k)$, can be derived by

$$\Pr(t_k) \equiv \sum_{i=1}^{n} \Pr(t_k|d_i) \cdot \Pr(d_i). \quad (10)$$

The above defined similarity formula can be written as:

$$sim(d_i, d_j) \simeq \frac{\Pr(d_i) \Pr(d_j)}{\sum_{l=1}^{m} f_{il} \sum_{l=1}^{m} f_{jl}} \sum_{k=1}^{m} \frac{f_{ik} f_{jk}}{\Pr(t_k)}. \quad (11)$$

If we represent document vector $\vec{d_i}$ of d_i by $tf \cdot idf$ weighting scheme

$$\vec{d_i} = (tf_{i1} \cdot idf_1, tf_{i2} \cdot idf_2, \ldots, tf_{im} \cdot idf_m) \quad (12)$$

and

$$\begin{aligned} tf_{ik} &= f_{ik}, & (13) \\ idf_k &= \frac{1}{\sqrt{\Pr(t_k)}}, & (14) \\ len_i &= \sum_{l=1}^{m} f_{il}, & (15) \end{aligned}$$

the similarity formula can be transformed as:

$$sim(d_i, d_j) = \Pr(d_i) \Pr(d_j) \frac{\vec{d_i} \cdot \vec{d_j}}{len_i \times len_j}. \quad (16)$$

4. Clustering algorithm based on *K*-means method

4.1. *K*-means clustering method

The *K*-means method [9] is one of the commonly used clustering methods in data mining and in topic detection and tracking [12]. Its general algorithm is:

1. Select *K* documents randomly and form initial *K* clusters.
2. Compute initial cluster representatives.
3. Remaining documents are compared with the cluster representatives and assigned to the most appropriate cluster.
4. If there is no change to cluster assignment result, terminate the procedure. Otherwise, re-compute the cluster representatives and return to step 3.

4.2. Clustering index

Our clustering algorithm introduces the *clustering index G*, which is computed by:

$$G \equiv \sum_{p=1}^{K} |C_p| \cdot avg\_sim(C_p), \quad (17)$$

where $|C_p|$ is the number of documents in cluster C_p, and $avg\_sim(C_p)$ is the average similarity of documents in cluster C_p and is defined as:

$$avg\_sim(C_p) \equiv \frac{1}{|C_p|(|C_p|-1)} \sum_{d_i \in C_p} \sum_{d_j \in C_p, d_i \neq d_j} sim(d_i, d_j). \quad (18)$$

$avg\_sim(C_p)$ is used as a measure to decide the goodness and poorness of a clustering result. $avg\_sim(C_p)$ is regarded as an *intra-cluster similarity*.

4.3. Proposed clustering algorithm

Our proposed algorithm is an extension of the *K*-means algorithm. The *K*-means algorithm is extended considering the characteristics of the similarity formula shown above. A document is allocated to a cluster such that the assignment makes the largest increase on the *intra-cluster similarity*.

- **Initial process**

1. Select K documents randomly and form initial K clusters.
2. Compute cluster representatives.
3. Compute the intra-cluster similarities and the clustering index G.

- **Repetition process**
  1. For each document d, do the following two steps:
     (a) For each cluster, compute the intra-cluster similarity when d is appended to the cluster.
     (b) Assign d to the cluster of which the increase of intra-cluster similarity is the largest. If any assignment does not increase the intra-cluster similarity, put d into the outlier list.
  2. Recompute cluster representatives.
  3. Recompute G and take it as G_{new} .
  4. If $(G_{\text{new}} - G_{\text{old}})/G_{\text{old}} < \delta$, terminate, where δ is a pre-defined constant.
  5. Return to Step 1.

Documents put in the outlier list are regarded as normal documents in the next iteration since the documents may not fall in the outlier list next time as contents of clusters will change.

Our extended K-means method introduces a clearer criterion for clustering convergence and the handling of outliers, those documents which are considered not relevant to other documents in the clustering dataset.

4.4. Efficient calculation using cluster representative

In the step 1 of the repetition process of our proposed clustering method, the computation overhead of the average similarity, avg_sim, shown in formula (18), for each cluster each time a document is removed or appended to the cluster, is very large. So we will show below the efficient computation of the avg_sim by using cluster representatives. This is an extended idea of Scatter/Gather [3].

If m is the total number of index term, vector of cluster representative $\vec{c_p}$ of cluster C_p is defined by

$$\vec{c_p} \equiv (c_1^p, c_2^p, \ldots, c_m^p). \tag{19}$$

For $1 \leq k \leq m$,

$$c_k^p \equiv \sum_{d_i \in C_p} \frac{\Pr(d_i) \cdot tf_{ik} \cdot idf_k}{len_i}. \tag{20}$$

The similarity between cluster representatives of cluster C_p and C_q is defined by

$$cr\_sim(C_p, C_q) \equiv \sum_{k=1}^{m} c_k^p c_k^q. \tag{21}$$

Then, the self similarity of cluster representative of cluster C_p, $cr\_sim(C_p, C_p)$, can be expanded as

$$\begin{aligned} & cr\_sim(C_p, C_p) \\ = \; & |C_p|(|C_p| - 1) \cdot avg\_sim(C_p) + ss(C_p). \end{aligned} \tag{22}$$

$ss(C_p)$ is the sum of the similarity of each document in cluster C_p with itself:

$$ss(C_p) = \sum_{d_i \in C_p} sim(d_i, d_i). \tag{23}$$

The average similarity in cluster C_p, $avg\_sim(C_p)$, can be written as

$$avg\_sim(C_p) = \frac{cr\_sim(C_p, C_p) - ss(C_p)}{|C_p|(|C_p| - 1)}. \tag{24}$$

If cluster $C_r = C_p \cup C_q$ and C_p, C_q have no same elements ($C_p \cap C_q = \emptyset$), then

$$\begin{aligned} & avg\_sim(C_r) \\ = \; & [cr\_sim(C_p, C_p) + 2cr\_sim(C_p, C_q) \\ & + cr\_sim(C_q, C_q) - ss(C_p) - ss(C_q)] \\ & /[(|C_p| + |C_q|)(|C_p| + |C_q| - 1)]. \end{aligned} \tag{25}$$

If C_q is a singleton cluster, that is $C_q = \{d_q\}$,

$$\begin{aligned} & avg\_sim(C_r) \\ = \; & \frac{cr\_sim(C_p, C_p) + 2cr\_sim(C_p, C_q) - ss(C_p)}{|C_p|(|C_p| + 1)}. \end{aligned} \tag{26}$$

That is, to compute the avg_sim of an existing cluster C_p when d_q is appended to the cluster, we need to compute the similarity of cluster representatives $cr\_sim(C_p, C_q)$ only since $cr\_sim(C_p, C_p)$, $ss(C_p)$ and $|C_p|$ are computed once when cluster C_p is created and can be used as many times as required in one clustering iteration. By using the formula (26), we can reduce the cost to re-compute avg_sim when a document is appended to a cluster.

We can formulate similar update formulas for deletion when a document is removed from a cluster. They are omitted for simplicity.

5. Incremental statistics update and clustering

In traditional document clustering, clustering is performed from scratch. However, since our target documents are on-line documents such as news articles which are delivered continually, we should take such dynamic nature into consideration.

Our novelty-based document clustering method incorporates the incremental statistics updating process and incremental clustering process as shown below.

5.1. Incremental statistics update

The values of some statistics and probabilities such as document weight dw_i, total weight of all documents tdw, the selection probability $Pr(d_i)$, etc., change with time and when new documents are incorporated into the document repository. Recalculating those statistics and probabilities from scratch tends to be costly for a large dataset. In our approach, the values of those statistics and probabilities are updated incrementally by using the values of previous statistics and probabilities to

achieve efficient updates. We will briefly introduce the incremental calculation of some values below. Details are described in [8].

Let the last update time of the given document set consisting of m documents $d_1, \ldots, d_m$ be $t = \tau$. Namely, the most recent documents are incorporated into the document set at $t = \tau$. Then suppose that m' new documents $d_{m+1}, \ldots, d_{m+m'}$ are appended at the time $t = \tau + \Delta\tau$. Therefore, their acquisition times are $T_{m+1} = \cdots = T_{m+m'} = \tau + \Delta\tau$. Let all the index terms contained in the document set at time $t = \tau$ be $t_1, \ldots, t_n$ and the additional terms incorporated by the insertion of documents $d_{m+1}, \ldots, d_{m+m'}$ be $t_{n+1}, \ldots, t_{n+n'}$.

1. Updating of dw_i's: First we consider the update of weights of documents $d_1, \ldots, d_m$. We have already assigned a weight $dw_i|_\tau$ to each document d_i $(1 \leq i \leq m)$ at the last update time $t = \tau$. These weights have to be updated to $dw_i|_{\tau+\Delta\tau}$ in this update time. Since the relationship

$$\begin{aligned} dw_i|_{\tau+\Delta\tau} &= \lambda^{\tau+\Delta\tau-T_i} \\ &= \lambda^{\Delta\tau} dw_i|_\tau \end{aligned} \tag{27}$$

holds between $dw_i|_\tau$ and $dw_i|_{\tau+\Delta\tau}$, we can easily derive $dw_i|_{\tau+\Delta\tau}$ from $dw_i|_\tau$ by simply multiplying $\lambda^{\Delta\tau}$ to $dw_i|_\tau$. This property for the efficient update is due to the selection of the exponential forgetting factor in our document forgetting model.

For the new incoming documents $d_{m+1}, \ldots, d_{m+m'}$, we simply set $dw_i|_{\tau+\Delta\tau} = 1$ $(m+1 \leq i \leq m+m')$.

2. Updating of tdw: For the total weight of all the documents tdw, we can utilize the following update formula:

$$\begin{aligned} tdw|_{\tau+\Delta\tau} &= \sum_{l=1}^{m+m'} \lambda^{\tau+\Delta\tau-T_l} \\ &= \lambda^{\Delta\tau} tdw|_\tau + m'. \end{aligned} \tag{28}$$

3. Calculation of $\Pr(d_i)$'s: $\Pr(d_i)$, the occurrence probability of document d_i, is given by

$$\Pr(d_i)|_{\tau+\Delta\tau} = \frac{dw_i|_{\tau+\Delta\tau}}{tdw|_{\tau+\Delta\tau}}. \tag{29}$$

Since we have already obtained $dw_i|_{\tau+\Delta\tau}$ and $tdw|_{\tau+\Delta\tau}$ in Step 1 and 2, we can easily calculate $\Pr(d_i)$ when it is required.

5.2. Incremental novelty-based document clustering

We use the following incremental clustering procedure. Let τ be the timestamp when the previous clustering was performed, and $\tau' = \tau + \Delta\tau$ be the current timestamp.

1. Incorporate new documents $d'_1, \ldots, d'_n$ arrived in the period $\tau \leq t \leq \tau'$ into the target document set.
2. Delete old documents d such that d satisfy $dw(d) < \epsilon$, where ϵ is a constant obtained from a user by specifying a *life span value* γ, the period that all documents in the document set are active. The value ϵ is derived as $\epsilon = \lambda^\gamma$. Then update statistics for all documents in the document set using the method shown above.
3. Perform clustering based on the proposed variant of the *K*-means clustering procedure shown above, but reuse the cluster representatives of the previous clustering performed at the timestamp τ and take them as the initial cluster representatives. The idea behind this is that the clustering tendency does not change greatly with minor modification to the target document set. Using the previous clustering results, we can accelerate the clustering process.

6. Experimentation

6.1. Experiment 1

The objective of this experiment is to evaluate the efficiency of our clustering method by comparing the computation time of the incremental version and non-incremental version of our method.

In this experiment, we use data in the original TDT2 corpus [1]. The corpus consists of chronologically ordered news articles obtained from 6 newswire sources and TV/radio broadcast services, including ABC, APW, CNN, NYT, PRI and VOA, from January 4th to June 30th 1998. This corpus consists of approximately 64,400 documents. The experiment is performed on a PC with Pentium 4 CPU, speed 3.2 GHz and 1 GB of RAM. The program is written using Ruby programming language. We use the parameter values K = 32, half life span β = 7 days and life span γ = 14 days. They correspond to $\lambda = 0.9$ and $\epsilon = 0.25$.

We execute the clustering method on the data from Jan 4th to Jan 18th (4,327 docs) using the non-incremental version of our approach. The statistics update takes 25min21sec and clustering takes 58min17sec. On the other hand, suppose that we already have the clustering results performed on Jan 4th to Jan 17th. Thus when we reuse the statistics and clustering results of the Jan 4th to Jan 17th, i.e., we process only the data on Jan 18th (205 docs), we can achieve statistics updating time 1min45sec and clustering time 15min25sec. Table 1 depicts the computation times just described.

Approach	Dataset	Statistics Updating	Clustering
Non-incremental	Jan4-Jan18	25min21sec	58min17sec
Incremental	Jan18	1min45sec	15min25sec

Table 1. Computation times of incremental and non-incremental approaches

We can say that using the incremental approach we can achieve efficient statistics update and clustering. In statistics update, the computation time is approximately proportional to the number of the documents to be updated. Hence, the incremental approach outperforms the non-incremental one since the number of documents to be updated is relatively small compared to the number of existing documents in the repository. For clustering, the computation time depends heavily on the

characteristics of the documents themselves and on the number of iterations. For incremental clustering, we think that a new cluster structure does not change much from the previous structure even if small number of documents are added and thus achieve faster computation time.

Although the incremental approach is obviously better in computation time, a question may arise concerning whether the incremental approach can provide similar clustering quality compared to the non-incremental approach. We will investigate this issue in future work.

6.2. Experiment 2

The main objective of our method is to answer to the question: "what are recent topics?". Recent topics should appear in the clustering results while old topics should not.

6.2.1. Experimental dataset The original TDT2 corpus as mentioned in Section 6.1 approximately consists of 64,000 documents from January 4th to June 30th 1998. However, there are only 11,201 documents are labeled with "YES" and/or "BRIEF" in the selected 100 topics (but 4 topics contain no documents) by the Linguistic Data Consortium [6]. In addition, we found that some documents among the annotated documents are marked with more than one label. Therefore, we selected documents which are marked with only one "YES" label and used in our experiments. The portion of TDT2 corpus which are marked with only one "YES" label consists of 7,578 documents corresponding to 96 topics dated from January 4th to June 30th 1998.

We split the selected 7,578 documents from January 4th to June 30th into 6 time windows. Each time window consists of news stories of 30 days, except for the last time window which consists of only 28 days. The first to sixth time windows consist of news stories for the period Jan4-Feb2, Feb3-Mar4, Mar5-Apr3, Apr4-May3, May4-Jun2 and Jun3-Jun30, respectively. The statistics of the divided time window is shown in Table 2, and some topics, including the number of documents in the topics and topic names, of the selected corpus are presented in Table 5.

	First	Second	Third	Fourth	Fifth	Sixth
No. of docs	1820	2393	823	570	1090	882
No. of topics	30	44	47	39	40	43
Min. topic size	1	1	1	1	1	1
Max. topic size	461	875	129	96	327	138
Med. topic size	16.5	6	4	5	4.5	4
Mean topic size	60.67	54.39	17.51	14.62	27.25	20.51

Table 2. Time window statistics for selected TDT2 subset

6.2.2. Experimental setting We apply our method to the time window data described in the previous section using non-incremental processing version of our proposed method. For this experiment, we do not use the incremental version as we have observed that the clustering results generated by the incremental and the non-incremental versions of our method are roughly close to each other. In addition, the incremental clustering is suited for on-line setting in which documents are delivered continuously. However, this experiment only requires the final result when we have processed all the documents in a time window. Therefore, batch-oriented non-incremental version is suited for this experiment. Furthermore, the purpose of this experiment is to evaluate the performance of our method based on different values of half life span parameters, and not to evaluate the performance of the incremental and the non-incremental approach.

To answer to the question "what are recent topics?", we perform two experiments. They are designed to observe clustering results using two different half life span values, 7 days and 30 days, which correspond to $\lambda = 0.9$ and $\lambda = 0.98$ respectively.

Table 5: Some Topics in selected TDT2 corpus from Jan4-Jun30 1998

Topic ID	Count	Topic Name
20001	1034	Asian Economic Crisis
20002	923	Monica Lewinsky Case
20004	19	McVeigh's Navy Dismissal & Fight
20005	38	Upcoming Philippine Elections
20011	18	State of the Union Address
20012	150	Pope visits Cuba
20013	530	1998 Winter Olympics
20014	2	African Leaders and World Bank Pres.
20015	1439	Current Conflict with Iraq
20017	17	Babbitt Casino Case
20018	99	Bombing AL Clinic
20019	110	Cable Car Crash
20020	32	China Airlines Crash
20021	53	Tornado in Florida
20022	30	Diane Zamora
20023	125	Violence in Algeria
20026	70	Oprah Lawsuit
20030	2	Pension for Mrs. Schindler
20031	36	John Glenn
20032	126	Sgt. Gene McKinney
20033	83	Superbowl '98
20036	5	Rev. Lyons Arrested
20039	119	India Parliamentary Elections
20040	6	Tello (Maryland) Murder
20041	26	Grossberg baby murder
20042	29	Asteroid Coming??
20043	15	Dr. Spock Dies
20044	277	National Tobacco Settlement
20046	5	Great Lake Champlain??
20047	93	Viagra Approval
20048	125	Jonesboro shooting
20062	2	Mandela visits Angola
20063	16	Bird Watchers Hostage
20064	11	Race Relations Meetings
20065	60	Rats in Space!
20070	415	India, A Nuclear Power?
20071	201	Israeli-Palestinian Talks (London)
20074	50	Nigerian Protest Violence
20075	7	Food Stamps
20076	225	Anti-Suharto Violence
20077	117	Unabomber
20078	15	Denmark Strike
20079	8	Akin Birdal Shot & Wounded
20082	4	Abortion clinic acid attacks
20083	17	World AIDS Conference
20085	8	Saudi Soccer coach sacked
20086	138	GM Strike
20087	79	NBA finals
20088	5	Anti-Chinese Violence in Indonesia
20096	64	Clinton-Jiang Debate
20097	2	Martin Fogel's law degree
20098	9	Cubans returned home
20099	1	Oregon bomb for Clinton?
20100	8	Goldman Sachs - going public?

The idea behind this selection is that the 7-day half life span assigns smaller weights to old documents and higher weights to recent ones in the clustering period. In the two experiments, we choose the same values for other parameters: K = 24 and life span = 30 days[1]. The 30-day life span will enable all documents to stay active during the clustering period since we use 30-day time window.

Moreover, a single forgetting factor value is applied to all documents in the selected dataset. We assume that all news articles have the same aging speed regardless of which topics an article is about. Choosing different forgetting factor values is unfeasible in our unsupervised learning clustering approach since we do not know in advance which topics a document belongs to before we group them and get clustering results. Moreover, The use of single half life span or forgetting factor enable the achievement of efficiency in incremental processing of our clustering approach.

6.2.3. Experimental results We evaluate the clustering results produced by our method using the following performance measures:

- Precision: $p = a/(a + b)$
- Recall: $r = a/(a + c)$
- $F_1 = 2rp/(r + p) = 2a/(2a + b + c)$

where a, b and c refer to the number of documents in each category in Table 3 respectively.

	On topic	Not on topic
In cluster	a	b
Not in cluster	c	d

Table 3. Distribution of documents

For the two clustering results, we compare our system generated clusters with the selected TDT2 topics and compute the precision and recall for each cluster. We determine a cluster is marked with a topic if the precision of the topic in the cluster is equal or greater than 0.60. If a cluster has no precision larger than 0.60, then the cluster is not marked with any topic. Figure 1 to Figure 4 are some examples of the precision and recall for the first and fourth time windows for the two half life span values.

We measure global performance of our method by microaverage F_1 and macroaverage F_1 [12]. F_1 is the harmonic mean of recall and precision. The *microaverage* F_1 is obtained by merging the Table 3 for each marked cluster by summing the corresponding cells and then using the merged table to produce global performance measures. The *macroaverage* F_1 is obtained by producing per-cluster performance measures, then averaging the corresponding measures [12]. The global performance by microaverage F_1 and macroaverage F_1 for the clustering results of the six time windows with the different half life span values β is shown in Table 4. Generally speaking, the long half life span $\beta = 30$ has better results than $\beta = 7$, but this is an expected result since the evaluation measure F_1 does not consider the novelty of topics. Since $\beta = 30$ resembles the conventional clustering, it shows better F_1 scores.

1 The life span parameter γ is defined by a user specifying the period that a document is active for clustering. It is used to define the value of the parameter ϵ in $\epsilon = \lambda^{\gamma}$. See [8] for details.

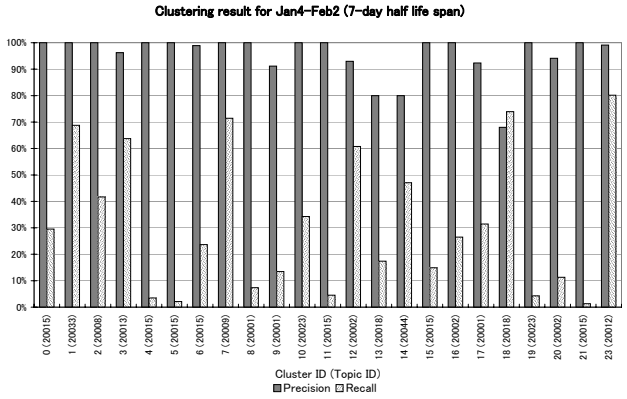

Figure 1. Clustering result for Jan4-Feb2 with 7-day half life span

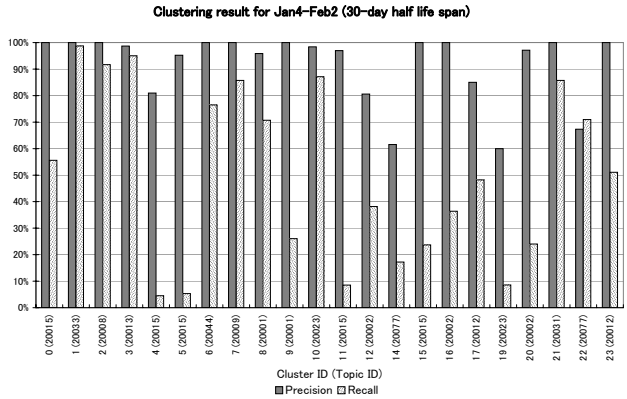

Figure 2. Clustering result for Jan4-Feb2 with 30-day half life span

What we want to investigate in this paper is whether clustering of short half life span $\beta = 7$ can detect "hot" topics and forget obsolete topics. We examine the properties of clustering results of the two different half life span values and how they answer to our question. Generally, recent topics appear in the clustering results of the 7-day half life span while old topics in the clustering period do not. In addition, some topics which have small number of documents but are recent topics are presented in the 7-day half life span clustering results but not presented in the clustering results of 30-day half life span. We will show some examples to prove this.

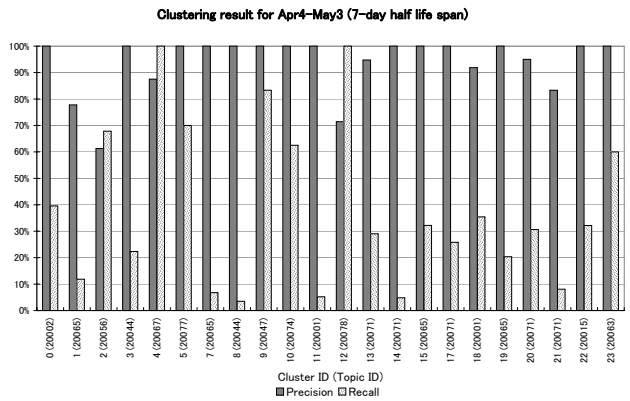

Figure 3. Clustering result for Apr4-May3 with 7-day half life span

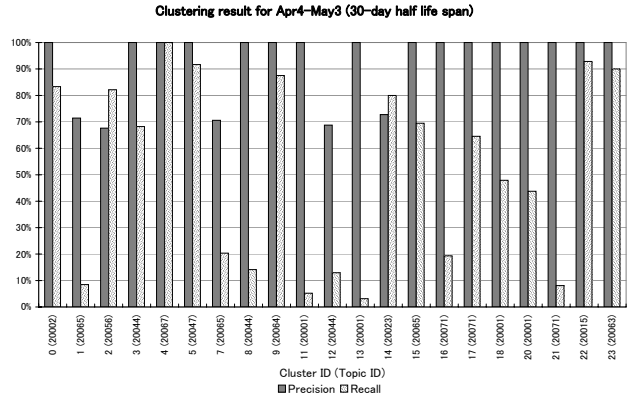

Figure 4. Clustering result for Apr4-May3 with 30-day half life span

For example, topic 20074 "Nigerian Protest Violence" as shown by the histogram in Figure 5, the topic scattered in the histogram, but slightly more densely in the fourth (Apr4-May3) and sixth (Jun3-Jun30) time window. The topic appears in the clustering of 7-day half life span in the fourth time window as the topic occurred quite recently in the period, but not in the clustering of the 30-day one. In the sixth time window, the topic appeared quite early in the period, so the clustering of 7-day half life span does not detect the topic but the 30-day one does.

Another example is topic 20077 about "Unabomber". As shown by its histogram in Figure 6, the topic occurred at the first half of the first time window (Jan4-Feb2) and disappeared and then emerged again late of the fourth time window (Apr4-May3). In the Jan4-Feb2 time window clustering result, the topic appears in the clustering with 30-day half life span, but not in the 7-day one. But in the Apr4-May3 time window, the topic is captured in a small cluster with 7 documents in the clustering result of 7-day half life span, but is not generated

Time window	Microaverage F_1	Macroaverage F_1
first ($\beta = 7$ / $\beta = 30$)	0.34 / 0.52	0.42 / 0.59
second ($\beta = 7$ / $\beta = 30$)	0.40 / 0.55	0.50 / 0.67
third ($\beta = 7$ / $\beta = 30$)	0.32 / 0.53	0.37 / 0.61
fourth ($\beta = 7$ / $\beta = 30$)	0.39 / 0.53	0.48 / 0.59
fifth ($\beta = 7$ / $\beta = 30$)	0.39 / 0.53	0.50 / 0.57
sixth ($\beta = 7$ / $\beta = 30$)	0.51 / 0.60	0.55 / 0.66

Table 4. Microaverage F_1 and macroaverage F_1 for clustering results of the six time windows with different half life span values

by the 30-day one as the topic occurred recently in the Apr4-May3 time window. Even if the number of documents for the topic in the period is very small (10 documents), the clustering of 7-day half life span can still detect the topic.

Topic 20078 "Denmark Strike" happened in late of the 4th clustering period Apr4-May3 and early of the 5th one May4-Jun2, all in small number of occurrences. The histogram of the topic is shown in Figure 7. In the Apr4-May3 clustering period, the clustering of 7-day half life span generated the topic impressively with value one for recall and high precision while the clustering of the 30-day one can not.

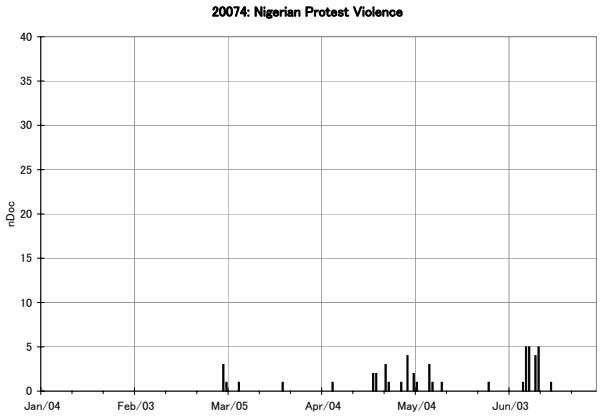

Figure 5. Histogram for topic 20074

As another observation, we can notice that sizes of topics of the dataset we used in the experiments fluctuate wildly between small and large topics, so our clustering method generally groups documents of those large topics into several clusters. For example, topic 20001 "Asian Economic Crisis" (histogram shown in Figure 8) and topic 20002 "Monica Lewinsky Case" (histogram shown in Figure 9) appears in more than one clusters in the clustering results of the first time window (Figure 1 and Figure 2).

In summary, the clustering of 30-day half life span produces better results with higher precision and recall than the one with 7-day half life span as shown in some figures of the recall and precision and the averaged measures in Table 4. We can say that if one wants a clustering with larger precision and recall without the notion of novelty, 30-day half life span or larger half life span values are a good choice. However, if one wants a clus-

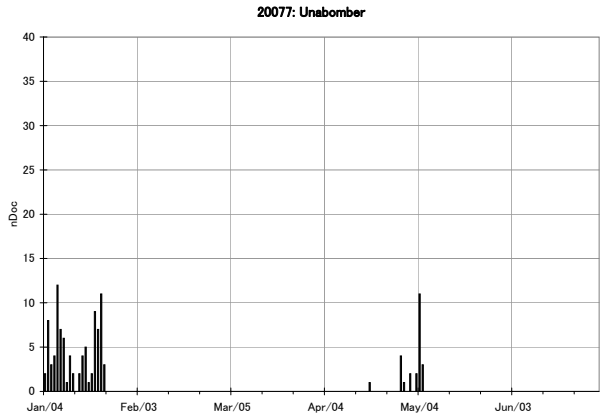

Figure 6. Histogram for topic 20077

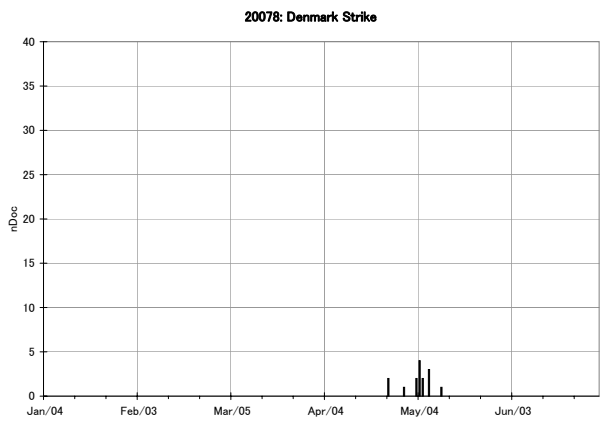

Figure 7. Histogram for topic 20078

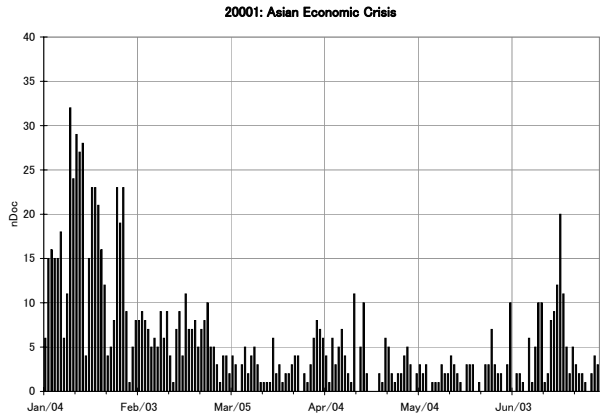

Figure 8. Histogram for topic 20001

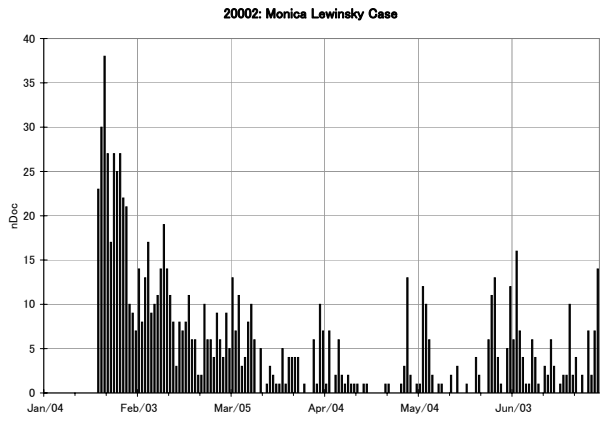

Figure 9. Histogram for topic 20002

tering which produces recent hot topics, then smaller half life span values such as 7 days are superior.

7. Conclusions and future work

This paper proposed a new clustering algorithm for the novelty-based incremental document clustering method and showed the comparison of computation time of the incremental and the non-incremental version of our method by proving the incremental version is more efficient than the non-incremental one. We also showed clustering results using two different small and large half life span values. Clustering of both parameter values settings generally yield high precision and recall. The clustering using a small half life span value is noticeably better in generating clusters of recent topics while the one of larger value performs well in general setting in which the novelty of a topic is not considered.

As future work, we will show that the incremental and the non-incremental version of our method produce similar clustering results. Future work also includes a method to estimate the appropriate K value and experiments using the small and large forgetting factor values on larger time window size to analyze the properties of the method.

Acknowledgement

This research is partly supported by the Grant-in-Aid for Scientific Research (16500048) from Japan Society for the Promotion of Science (JSPS), Japan, and the Grant-in-Aid for Scientific Research on Priority Areas (16016205) from the Ministry of Education, Culture, Sports, Science and Technology (MEXT), Japan. In addition, this work is supported by the grants from the Asahi Glass Foundation and the Inamori Foundation.

References

[1] J. Allan (ed.): *Topic Detection and Tracking: Event-based Information Organization*, Kluwer, 2002.

[2] F. Can: "Incremental Clustering for Dynamic Information Processing", *ACM TOIS*, Vol. 11, No. 2, pp. 143–164, 1993.

[3] D. Cutting, D.R. Karger, J.O. Pedersen, and J.W. Tukey: "Scatter/Gather: A Cluster-based Approach to Browsing

Large Document Collections", *Proc. of 15th ACM SIGIR Conference*, , pp. 318–329, Jun. 1992.

[4] W.B. Frakes, and R. Baeza-Yates: "Information Retrieval: Data Structures and Algorithms", Prentice Hall PTR, 1992.

[5] J. Han, and M. Kamber: "Data Mining: Concepts and Techniques", Morgan Kaufmann Publishers, 2001.

[6] http://www.ldc.upenn.edu/

[7] http://www.nist.gov/speech/tests/tdt/

[8] Y. Ishikawa, Y. Chen, and H. Kitagawa: "An On-line Document Clustering Method Based on Forgetting Factors", *Proc. of 5th European Conference on Research and Advanced Technology for Digital Libraries (ECDL'01)*, pp. 325–339, 2001.

[9] A.K. Jain, M.N. Murty, and P.J. Flynn: "Data Clustering: A Review", *ACM Computing Surveys*, 31(3), 1999.

[10] K.R. McKeown, R. Barzilay, D. Evans, V. Hatzivassiloglou, J.L. Klavans, A. Nenkova, C. Sable, B. Schiffman, and S. Sigelman: "Tracking and Summarizing News on a Daily Basis with Columbia's Newsblaster", *Proc. of Human Language Technology Conference (HLT'02)*, 2002.

[11] D. Radev, J. Otterbacher, A. Winkel, and S. Blair-Goldensohn: "NewsInEssence, Summarizing Online News Topics", *Proc. of Communications of the ACM*, pp. 95–98, 2005.

[12] Y. Yang, J.G. Carbonell, R.G. Brown, T. Pierce, B.T. Archibald, and X. Liu: "Learning Approaches for Detecting and Tracking News Event", *IEEE Intelligent Systems*, Vol. 14, No. 4, 1999.