Incremental Clustering Based on Novelty of On-line Documents – Experiments and Evaluation

Sophoin KHY[†], Yoshiharu ISHIKAWA[††,†††], and Hiroyuki KITAGAWA[††,†††]

† Master's Program in Science and Engineering
†† Graduate School of Systems and Information Engineering
††† Center for Computational Sciences
University of Tsukuba
Tennoudai 1–1–1, Tsukuba, Ibaraki, 305–8573 Japan

Abstract Due to increased requirement of managing and extracting useful information from myriad data, clustering has been widely used as a fundamental technique in data mining and as a preprocessing step for other algorithms such as characterization and classification. Various clustering researches have been conducted since decades ago. In our research about clustering, a novelty-based incremental document clustering method which takes into consideration novelty of documents in a similarity measure and performs clustering based on an extended algorithm of the K-means method was proposed. This paper further examines the performance of the incremental and non-incremental processing of the clustering method and effect of parameter values on the behavior of the method by showing experimental results using various values of parameters.

Key words document clustering, novelty, incremental processing, on-line documents

1. Introduction

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 and the requirement for managing and extracting useful information from myriad electronic documents is on the rise.

Document clustering has been used as a core technique in managing vast amount of data and providing needed information. It is a method which groups similar documents into same clusters [2], [3], [10]. 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 [11], [12].

In on-line environments, users are apt to be interested in new and up-to-date information. Traditional clustering focuses on grouping similar documents into clusters by treating each document with equal weight. Our research focuses on a clustering method which has biases on recent documents, novelty-based document clustering. The novelty-based document clustering is a clustering method which takes into consideration novelty of documents in a similarity measure and performs clustering based on an extended algorithm of the K-means method.

This paper examines the behavior of the clustering method by conducting two experiments. The first experiment is aimed at evaluating the efficiency and effectiveness of the incremental and the non-incremental processing of the clustering method. The second experiment is for investigating the effect of parameters on the clustering method and appropriate K values, the number of clusters in the K-means method, for different parameter values. The results of these experiments will be shown and the impact of parameter values on the behavior of the clustering method will be discussed.

We start in Section 2 by reviewing related work. Section 3 describes the similarity measure. In Section 4 we introduce our proposed extended K-means method. Section 5 presents experiments and evaluation. Section 6 concludes the paper and describes future work.

2. Related Work

2.1 Topic Detection and Tracking

Topic Detection and Tracking (TDT) is a research project organized by NIST [7]. TDT addresses multiple sources of information including 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. Generally, there are five 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,
- Link detection - Detect whether or not two stories are topically linked.

Clustering 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 that each 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 cannot 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.

Yang et al. proposed a method called GAC (group-average clustering) [13] 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 the 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 [13]. 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 a linear decaying-weight function is incorporated in the similarity function.

Compared with GAC, our clustering 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 incorporates a time window and a linear decaying weight in the similarity function, our method uses an exponential decaying factor by which the weight of a document decreases exponentially. High weights are assigned to recent documents and low weights to old ones. Our objective is not only to generate clusters but also to reflect current trend of hot or recent topics.

F^{2}ICM (Forgetting-Factor-based Incremental Clustering Method) [8] is a novelty-based document clustering method partially based on C^{2}ICM [2]. F^{2}ICM uses the clustering algorithm proposed in C^{2}ICM and devises a similarity measure based on an exponential decaying factor. In the clustering algorithm, F^{2}ICM first computes the seeds from documents and then classifies documents sequentially based on the seeds. The novelty-based incremental document clustering method used in this paper uses the forgetting-factor-based similarity function proposed in F^{2}ICM, but introduces a clustering algorithm extending the K-means algorithm. The extended K-means clustering algorithm will be described in detail in Section 4.2.

3. Similarity Measure

In this section, we briefly provide an overview of the forgetting-factor-based similarity function introduced in F^{2}ICM [8]. The similarity measure is derived from the *document forgetting model*. 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 $\lambda^{\beta} = 1/2$. Then, 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:

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

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:

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

where t_k is an index term. The co-occurrence probability of document d_i and d_j is:

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

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

$$sim(d_i,\, d_j) \equiv \Pr(d_i,\, d_j). \tag{7}$$

This definition is based on a probabilistic model that the similarity scores between document d_i and d_j is a process of searching for two documents in a document set such that the two documents are similar to each other by having some common terms.

From the 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

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

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

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

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

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

4. Clustering Algorithm based on *K*-means Method

4.1 Clustering Index

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

$$G \equiv \sum_{p=1}^{K} |C_p|\cdot avg\_sim(C_p), \tag{11}$$

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:

$$avg\_sim(C_p) \equiv \frac{1}{|C_p|(|C_p|-1)} \sum_{d_i \in C_p} \sum_{d_j \in C_p,\, d_i \neq d_j} sim(d_i,\, d_j). \tag{12}$$

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

4.2 Proposed Clustering Algorithm

The K-means method [10] is one of the commonly used clustering methods in data mining and in topic detection and tracking [13]. The clustering algorithm used in this paper is an extension of the K-means algorithm. A document is allocated to a cluster such that the assignment causes 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 intra-cluster similarities and 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 a cluster such that the increase of the intra-cluster similarity is the largest. If any assignment does not increase the intra-cluster similarity, put d into an 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 they may not fall in the outlier list again next time since contents of clusters change.

Our extended K-means method introduces a clearer criterion for clustering convergence and handling of outliers. The algorithm shown above is a brief summary of the full algorithm [9] which contains more features for efficient computation.

Our novelty-based document clustering method incorporates the incremental statistics updating process and incremetal clustering process [8], [9] to accelerate the processes.

5. Experimentation

5.1 Dataset

The TDT2 corpus developed by the Linguistic Data Consortium [6] consists of chronologically ordered news articles obtained from newswire sources and TV/radio broadcast services. The corpus consists of 64,398 documents from January 4th to June 30th 1998. However, there are only 11,201 documents labeled with topics (96 topics) as "YES" and/or "BRIEF". In addition, we found that some documents among the annotated documents are marked with more than one label. Therefore, we selected only those documents marked with only one "YES" label and used in our experiments. There are 7,578 documents corresponding to 96 topics dated from January 4th to June 30th 1998 obtained by this selection. These TDT2 subset is called "selected TDT2 corpus". Some topics in the selected TDT2 corpus are presented in Table 1.

Table 1 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
20011	18	State of the Union Address
20012	150	Pope visits Cuba
20013	530	1998 Winter Olympics
20015	1439	Current Conflict with Iraq
20018	99	Bombing AL Clinic
20026	70	Oprah Lawsuit
20033	83	Superbowl '98
20044	277	National Tobacco Settlement
20076	225	Anti-Suharto Violence
20077	117	Unabomber
20078	15	Denmark Strike
20082	4	Abortion clinic acid attacks
20083	17	World AIDS Conference
20086	138	GM Strike
20087	79	NBA finals
20088	5	Anti-Chinese Violence in Indonesia
20096	64	Clinton-Jiang Debate
20099	1	Oregon bomb for Clinton?
20100	8	Goldman Sachs - going public?

5.2 Evaluation Measure

Clustering results are evaluated by the following performance measures [13]:

- 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 2 respectively.

Table 2 Distribution of documents

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

For each clustering result, the system generated clusters are compared with the selected TDT2 topics and the precision and recall for each cluster are computed. Based on several observations, we define a cluster is marked with a topic if the precision of the topic in the cluster is equal to or greater than 0.60. If a cluster has no precision larger than 0.60, then the cluster is not marked with any topic.

Then we measure the global performance of our method by microaverage F_1 and macroaverage F_1 [13]. F_1 is a harmonic mean of recall and precision. *Microaverage* F_1 is obtained by merging the Table 2 for each marked cluster by summing the corresponding cells and then using the merged table to produce global performance measures. *Macroaverage* F_1 is obtained by producing per-cluster performance measures, then averaging the corresponding measures [13].

Since the goal of our clustering method is to generate clusters reflecting current trend of recent topics, it is necessary to identify what recent topics are. As TDT2 evaluation dataset do not provide novelty of topics, we introduce an evaluation concept which is used in Experiment 2 where 60-day time window is adopted. A topic is judged as *recent* (R) if the topic has at least two documents in the interval 51st-60th day. Or, if it has at least two documents in the interval 31st-50th day, it is judged as *less recent* (LR). Otherwise, it is considered *old* (O). For example, by the definition, by histogram in Figure 1, topic "Unabomber" is a *recent* topic in the Mar5-May3 time window and an *old* topic in Jan4-Mar4 and May4-Jun30 time windows. In the histogram in Figure 2, topic "NBA finals" is a *less recent* topic in the last May4-Jun30 time window.

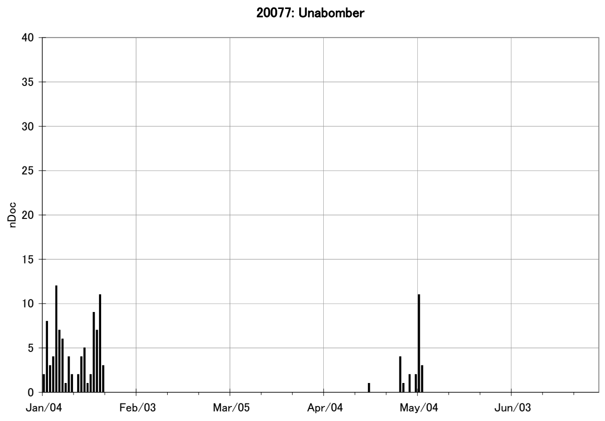

Figure 1 Histogram for topic 20077

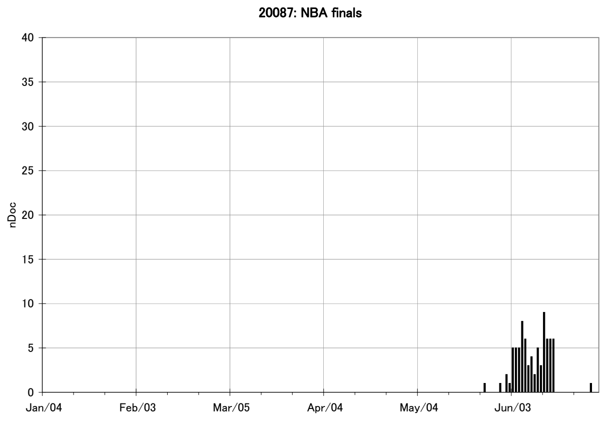

Figure 2 Histogram for topic 20087

5.3 Experiment 1

The objective of this experiment is to compare the efficiency and effectiveness of the incremental process with the non-incremental novelty-based clustering method. To this objective, computation time and cluster quality of the two processes will be assessed.

5.3.1 Experimental Dataset

The selected TDT2 dataset is split into six contiguous and non-overlapping 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 correspond to 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 3.

Table 3 Statistics for 30-day time window of selected TDT2 corpus

	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

5.3.2 Experimental Setting

In this experiment, the number of clusters $K = 24$, life span $\gamma = 30$ days, and two half life span values, $\beta = 7$ days and 30 days, are selected. These half life span values correspond to forgetting factor values $\lambda = 0.91$ and $\lambda = 0.98$ respectively. Choosing parameters with quite different values may provide clear insight into the effect of the half life span on the performance of the clustering method.

In addition, the life span parameter 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 as $\epsilon = \lambda^{\gamma}$. If $dw_i < \epsilon$, the document will be deleted from the document repository. The 30-day life span will enable all documents to stay active during the clustering period since the 30-day time window is used.

Moreover, the same forgetting factor value is applied to all documents in a series of the clustering operations. That is all news articles are assumed to have the same aging speed regardless of which topics the articles are about. Choosing different forgetting factor values is not feasible in this clustering approach since we do not know in advance which topics a document belongs to before we group them and get clustering results. Moreover, use of the same forgetting factor enables the achievement of efficiency in incremental processing of the clustering approach.

5.3.3 Experimental Framework

The experimental framework is designed as follows.

(1) Non-incremental process: The six time window dataset described in the previous section is used as an input to this process.

(2) Incremental process: This approach is commonly adopted in the real world setting in which a system has already accumulated data and generated clusters. Then the incremental process is used to incrementally update the clustering results when new documents are continually delivered to the system. By modeling this procedure, the non-incremental clustering is performed on the first, Jan4-Feb2, time window data described in the previous section as a preliminary step for this experiment. After the preliminary clustering, the incremental process is adopted. The data in the selected TDT2 corpus from February 3rd to June 30th are incrementally and continually given as three-day input data to the clustering system using the incremental process which consists of incremental statistics update process and incremental clustering process.

(3) For both processes, clustering is performed using two sets of parameters:

- half life span = 7 days, life span = 30 days and $K = 24$,
- half life span = 30 days, life span = 30 days and $K = 24$.

The reason behind the selection of the three-day input data for the incremental process is that the number of documents in the selected TDT2 corpus used in the experiment is small, 7,578 documents, and spans over a long period of time from January 4th to June 30th, 1998. Hence the number of documents contained in one day is very small. Therefore, three-day data is used.

5.3.4 Experimental Results and Evaluation

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 and implemented on Cygwin.

Evaluation of Efficiency

To evaluate the efficiency of the incremental and the non-incremental processes, the computation time of statistics update and clustering consumed by the two processes are compared.

Table 4 and Table 5 show the computation time in seconds required by the non-incremental and incremental processes for 7-day half life span and 30-day half life span respectively. The computation time for incremental process is the average computation time required by the incremental process to execute the three-day dataset in each time window. In the first column of the tables, *inc* stands for the incremental process and the *non-inc* is short for the non-incremental one.

Table 4 Computation time of 7-day half life span (seconds)

Dataset	Statistics Updating	Clustering
Feb3-Mar4 (inc / non-inc)	135 / 1585	581 / 939
Mar5-Apr3 (inc / non-inc)	93 / 698	383 / 217
Apr4-May3 (inc / non-inc)	48 / 535	89 / 220
May4-Jun2 (inc / non-inc)	69 / 917	172 / 499
Jun3-Jun30 (inc / non-inc)	63 / 712	180 / 337

Table 5 Computation time of 30-day half life span (seconds)

Dataset	Statistics Updating	Clustering
Feb3-Mar4 (inc / non-inc)	133 / 1594	451 / 913
Mar5-Apr3 (inc / non-inc)	89 / 674	265 / 239
Apr4-May3 (inc / non-inc)	49 / 536	80 / 149
May4-Jun2 (inc / non-inc)	72 / 887	134 / 256
Jun3-Jun30 (inc / non-inc)	65 / 722	156 / 247

The non-incremental process takes thirty-day dataset, one time window, as its input, whereas the incremental process takes as its input three-day dataset a time. The evaluation of the efficiency of both processes here is not to compare the computation time of the process on thirty-day data with the process on three-day data. The idea is that rather than performing clustering from scratch every time new documents are recieved, adopting the incremental process, efficient execution time can be achieved.

The tables suggest that using the incremental process, in general, we can achieve faster statistics update and clustering time. In statistics update, the computation time is approximately proportional to the number of the documents to be updated. Since the number of documents to be updated by the incremental process is relatively small compared to the number of documents to be processed by the non-incremental process, hence, the incremental process is more efficient than the non-incremental one. For clustering, the computation time depends heavily on the characteristics of the documents themselves and on the number of iterations. For incremental clustering, a new cluster structure is thought to not change much from the previous structure even if small number of documents are added and thus achieves faster computation time.

Evaluation of Effectiveness

To assess the quality of clusters produced by the incremental and the non-incremental processes, clustering results are compared with the selected TDT2 topics and the precision and recall and the macroaverage F_1 and microaverage F_1 are computed.

Figure 3 and Figure 4 show the macroaverage F_1 and microaverage F_1 scores of the incremental and the non-incremental clustering results at each specific date using 7-day half life span and 30-day half life span respectively. In the figure, the dates on the x-axis are the dates that the clustering results are observed.

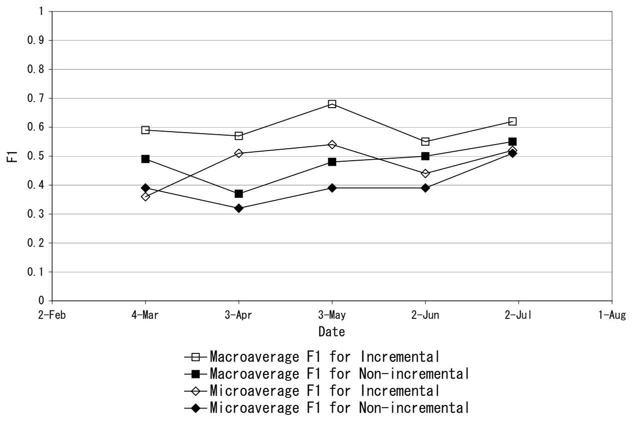

Figure 3 F1 scores for 7-day half life span

These results show that the quality of clusters of the incremental process is generally better than the non-incremental process. The non-incremental approach takes thirty-day dataset, one time window, as its input and processes them once. However, the incremental process takes as its input three-day dataset a time. Thus it takes ten times for the incremental approach to process data as much as the non-incremental one. Each clustering of the ten times probably gradually optimizes the association between documents in the clusters and results in better clustering results in the incremental process.

5.4 Experiment 2

The objective of the experiment is to examine the effect

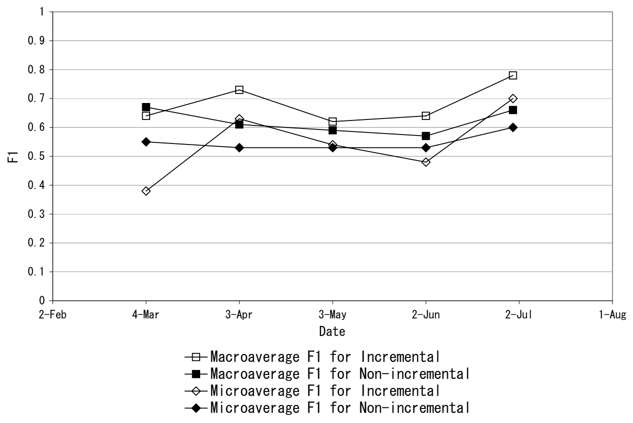

Figure 4 F1 scores for 30-day half life span

of papameters on the clustering method and to investigate appropriate K values for different values of half life span parameter.

5.4.1 Experimental Dataset

The selected TDT2 corpus is divided into three contiguous and non-overlapping time windows, Jan4-Mar4, Mar5-May3 and May4-Jun30. Each time window consists of news stories of 60 days, except for the last time window which has only 58 days. The statistics of the time windows is shown in Table 6.

Table 6 Statistics for 60-day time window of selected TDT2 corpus

	Jan4-Mar4	Mar5-May3	May4-Jun30
No. of docs	4213	1393	1972
No. of topics	51	61	54
Min. topic size	1	1	1
Max. topic size	1251	189	414
Med. topic size	16	5	8
Mean topic size	82.61	22.84	36.52

5.4.2 Experimental Setting

In this experiment, we selected two half life span values, 7 days and 60 days, which correspond to forgetting factor values $\lambda = 0.91$ and $\lambda = 0.99$ respectively. 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 a time window, while the 60-day one does not bias greatly towards recent topics. That is to say, the 7-day half life span causes weights of documents decayed from 0.91 to 0.003 and the 60-day one from 0.99 to 0.5 in 60 days. In addition, the life span parameter is set to the same value, 60 days, for each forgetting factor. Furthermore, a single forgetting factor value is applied to all documents in the selected dataset. The idea behind the selection is explained in Experiment 1.

5.4.3 Experimental Results and Evaluation

To select appropriate parameter values K for a half life span, first, we choose various K value, K = 8, 12, 16 and 32, for 7-day half life span, and K = 16, 24, 32 and 40, for 60-day half life span, and implement our method on the Jan4-Mar4 time window of the selected TDT2 dataset. The motivation behind the selection of smaller values of K for the 7-day half life span and larger values of K for the 60-day half life span is that for the 7-day half life span, as explained in the previous section, weights of obsolete documents are nearly zero and so are their similarity scores with other documents. Those documents are almost impossible to join a cluster. In other words, they are almost inactive.

In the experiments, we use the non-incremental process of our method as the purpose of the experiments is to define appropriate parameter values for our clustering context. The experiments only require the final results when we have processed all the documents in a time window. Therefore, batch-oriented non-incremental version is suited for the experiments.

The clustering results of the Jan4-Mar4 time window by different values of parameter K and evaluated by F_1 measures and novelty evaluation are shown in Table 7 for 7-day half life span and in Table 8 for 60-day half life span.

Table 7 Clustering results for 7-day half life span for Jan4-Mar4 time window

Topic ID	$K = 8$	$K = 12$	$K = 16$	$K = 32$	Recent?
20001	1	1	1	3	R
20002	1	1	2	2	R
20008				1	LR
20013	1	2	2	4	R
20015	1	1	3	6	R
20018		1	1	1	R
20020			1	1	R
20021	1	1	1	1	R
20022				1	LR
20023				1	R
20024				1	LR
20026	1	1	1	1	R
20032	1	1	1	1	R
20039	1	1	1	2	R
20040				1	R
20044			1	1	R
#of clusters	8	10	15	28	
Macro F1	0.50	0.45	0.42	0.35	
Micro F1	0.42	0.35	0.29	0.19	

Table 8 Clustering results for 60-day half life span for Jan4-Mar4 time window

Topic ID	$K = 16$	$K = 24$	$K = 32$	$K = 40$	Recent?
20001	1	4	5	5	R
20002	2	3	4	4	R
20004			1	1	O
20007				1	O
20008			1	1	LR
20009			1	1	LR
20012	1	1	1	1	LR
20013	2	2	3	3	R
20015	1	2	4	7	R
20018	1	1	1	1	R
20019	1	1	1	1	LR
20021			1	1	R
20022	1	1	1	1	LR
20023	1	1	1	1	R
20024			1	1	LR
20026	1	1	1	1	R
20031		1		1	LR
20032	1	1	1	1	R
20039	1	1	1	2	R
20044		1	1	1	R
20077	1	1		1	O
# of clusters	15	22	30	37	
Macro F1	0.81	0.66	0.63	0.59	
Micro F1	0.82	0.56	0.44	0.36	

As shown in Table 7 and Table 8, we see that topics detected by 7-day half life span are mostly recent and some are less recent, whereas the 60-day half life span detects recent, less recent and old topics.

For 7-day half life span, topics generated by $K = 8$ and $K = 12$ are all recent topics and the number of a single topic detected in many different clusters are less than $K = 16$ and $K = 32$. In addition, the scores of macroaverage F_1 and microaverage F_1 are higher than $K = 16$ and $K = 32$. Therefore we investigate the behavior of the clustering method further by implementing it on the other two time windows of the selected TDT2 dataset using $K = 8$ and $K = 12$ for the 7-day half life span. The results are shown in Table 9 for Mar5-May3 time window and Table 10 for May4-Jun30 time window.

Table 9 Clustering results for 7-day half life span for Mar5-May3 time window

Topic ID	$K = 8$	$K = 12$	Recent?
20001		1	R
20002	1	1	R
20015	1	1	R
20044		1	R
20047	1	1	R
20065	1	1	R
20067		1	R
20071	1	1	LR
# of clusters	5	8	
Macro F1	0.52	0.45	
Micro F1	0.41	0.33	

Table 10 Clustering results for 7-day half life span for May4-Jun30 time window

Topic ID	$K = 8$	$K = 12$	Recent?
20001	1	1	R
20002	1	2	R
20023	1	1	R
20044		1	R
20083	1	1	R
20086	1	1	R
20087	1	1	LR
20093		1	R
20096	1	1	R
# of clusters	7	10	
Macro F1	0.71	0.64	
Micro F1	0.64	0.57	

These results show that $K = 8$ produces higher F_1 values than $K = 12$, but $K = 12$ detects more recent topics than $K = 8$.

The clustering result of 60-day half life span in the Jan4-Mar4 time window (Table 8) shows the macroaverage F_1 and microaverage F_1 of $K = 16$ are relatively high compared with other K's. However, we think that this K value is too small to enable more topics to be generated since weights of the oldest documents in the 60-day half life span are 0.5 or higher as mentioned in the previous section which means they are still 'active' enough. Thus $K = 16$ are not considered further. For $K = 40$, this value is too large and may result in duplicated topics in different clusters and will not be considered here. Hence, we examine performance of $K = 24$ and $K = 32$ on the other two time windows, Mar5-May3 and May4-Jun30. The results are shown in Table 11 and Table 12.

Similar to the results of 7-day half life span, these results show that $K = 24$ generates clusters of higher values of F_1 than $K = 32$, but $K = 32$ generally detects more recent topics than $K = 24$.

5.4.4 Discussion

The 7-day half life span clustering detects mostly recent topics whereas the 60-day half life span one detects recent, less recent and also old topics. The 60-day half life span clustering generates higher F_1 scores than the 7-day half life span

Table 11 Clustering results for 60-day half life span for Mar5-May3 time window

Topic ID	$K = 24$	$K = 32$	Recent?
20001	3	2	R
20002	1	1	R
20015	1	3	R
20019	1	1	LR
20023	1	1	R
20024	1		O
20032	1	1	O
20039	1	1	O
20042	1	1	O
20044	2	3	R
20047	1	1	R
20048	1	2	LR
20056	1	1	R
20063		1	R
20065	2	2	R
20067		1	R
20071	2	2	R
20076		1	R
20077		1	R
# of clusters	20	26	
Macro F1	0.72	0.68	
Micro F1	0.70	0.63	

Table 12 Clustering results for 60-day half life span for May4-Jun30 time window

Topic ID	$K = 24$	$K = 32$	Recent?
20001	3	3	R
20002	3	5	R
20004	1		LR
20005		1	O
20008	1		LR
20015		1	R
20019	1		LR
20023	1		R
20044	1	1	R
20047		1	R
20070	1	3	R
20071	1	2	LR
20072		1	LR
20074	1	1	LR
20076	1	1	LR
20077	1	1	O
20082		1	O
20086	2	2	R
20087	1	1	LR
20091	1	1	R
20093	1	1	R
20096	1	2	R
20098	1		O
# of clusters	23	29	
Macro F1	0.77	0.66	
Micro F1	0.74	0.57	

one. This is because 7-day half life span results in a steep decrease of document weights and causes some documents handicapped to join a cluster. In addition, F_1 measure does not consider 'novelty' as in our clustering context. Evaluating our method by using F_1 measure results in low values of F_1 for small half life span values since many documents are not active in the clustering process, but are used to compare with the documents in the evaluated dataset.

Another observation of the results is that smaller K produces less number of topics but has higher F_1 values. For these experiments, if a user is interested in the values of F_1 measure, then $K = 8$ for 7-day half life span and $K = 24$ for 60-day half life span may be suitable for his/her requirement. On the other hand, if a user focuses on more topics than the the values of F_1, then $K = 12$ for 7-day half life span and $K = 32$ for 60-day half life span may be better.

6. Conclusions and Future Work

In this paper, we have shown that the incremental process is more efficient and effective than the non-incremental process. In addition, smaller half life span clustering performs better in detecting recent topics while larger one performs well in general setting in which novelty of a topic is not considered. Small Ks are more suitable for our clustering context than larger ones as they improve the effectiveness.

Future work includes a method to automatically estimate the appropriate K values for different values of half life span parameters and investigation of an evaluation measure that is more suitable for our novelty-based clustering context than the recall, precision and F_1 measures.

Acknowledgements

This research is partly supported by the Grant-in-Aid for Scientific Research (15300027 and 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] Y. Ishikawa, and H. Kitagawa: "An Improved Approach to the Clustering Method Based on Forgetting Factors", *DBSJ Letters*, Vol. 2, No. 3, pp. 53–56, December 2003.

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

[11] 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.

[12] 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.

[13] 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.