COMMUNICATIONS IN INFORMATION AND SYSTEMS
Vol. 4, No. 4, pp. 273-288, 2004

001

ESTIMATING TIMESTAMP FROM INCOMPLETE NEWS CORPUS*

HIROSHI UEJIMA[‡†], TAKAO MIURA[‡], AND ISAMU SHIOYA[§]

Abstract. Recently there have been a lot of researches for summarizing news stream and for detecting *edges* of new events in the news stream. But, in these tasks, all data are assumed to carry *timestamp* (temporal information). It is noteworthy that news articles without timestamp can't make any contribution to these tasks. In this investigation, we propose a new technique to estimate timestamps to any news articles using small number of incomplete news corpus. Here we learn temporal information and topic information by means of both EM algorithm and incremental clustering, then we estimate timestamp of news article based on events that are discussed in news corpus. In this work, we examine TDT2 corpus and we show how well our approach works by some experiments.

Key words: Timestamp estimation, TDT, Stream data, EM algorithm, Document Clustering

1. introduction.

Recently we have many chances to see news articles supplied constantly through internet from multiple sources. This environment drives many researchers to put much attention on grasping their contents easily and quickly.

One of the typical approaches is *Topic Detection and Tracking* (TDT)[8]. In TDT research project, several discussions have been investigated and proposed for the purpose of extracting semantic structure of topics automatically from document streams. TDT2004 project consists of 4 tasks: (1) New event detection, (2) Story link detection, (3) Topic detection, and (4) Topic tracking.

In TDT activity, it is well-known that temporal clustering works very well with event detection[1], that is, an event gets used to correspond to temporal cluster. We call these information *temporal documents*.

In these tasks, all the data are assumed to carry temporal information explicitly or implicitly and document without timestamp (or with incorrect timestamp) can't contribute to these tasks. When we get document stream from multiple sources, we have to examine the difference of timestamp between sources (like "scoop", "monthly magazine" and "anthology").

In this investigation we discuss how to estimate timestamp to a news article. Very often we miss correct timestamps to articles which discuss about past events or are supplied by other news sources. If we estimate timestamps correctly of the articles and re-order them, we could capture topics or trends of articles more easily and smoothly, and obtain more powerful TDT capability by putting several news

*(Eds.) Wai Lam, Rui-song Ye, Haiying Wang, and Jun Zhang.

†Current Affiliation: CASIO Computer Co.Ltd., Japan.

‡Dept.of Elect.& Elect. Engr., HOSEI University, 3-7-2 KajinoCho, Koganei, Tokyo, 184–8584 Japan, TEL: +81-42-387-6196, FAX: +81-42-387-6122, E-MAIL: miurat@k.hosei.ac.jp

§Dept.of Management and Informatics, SANNO University, 1573 Kamikasuya, Isehara, Kanagawa 259–1197 Japan.

sources into consistent states.

We have proposed incremental timestamp estimation method to estimate timestamps based on events[12]. But, in this method, we assume that we get the complete information from training data and we have all the events that we have presumed in advance.

However this is not true in actual document stream. There are a lot of events that we can't get from the complete data, where there are many groups of small numbers of articles that don't correspond to events. Here we propose a new technique that deals with incomplete situation.

In this investigation, we assume 4 kinds of information (news articles). $D_{topic,time}$ means a collection of information to which we get timestamps and topics, D_{time} and D_{topic} collections of information to which we can get only timestamp and topic respectively. D_n means the one of none of timestamp nor topics. We will estimate timestamps of any information in D_n and D_{topic} by using *all* the collections $D_{topic,time}, D_{time}, D_{topic}$ and D_n as training data.

Here let us outline of algorithm we propose.

(1) We classify a D_{time} and D_n into topics by learning topic information from $D_{topic,time}$, D_{topic} based on EM algorithm.
(2) Based on the results of 1), we make clustering from $D_{topic,time}$ and D_{time} into temporal events by single pass clustering.
(3) Based on results of 1) and 2), we assign each article of D_{topic} and D_n to the most suitable cluster with its topic information.
(4) Finally we extract timestamp values from clusters and put them to articles of D_{topic} and D_n.

EM algorithm stands for Expectation-Maximization algorithm. As the name shows, we can estimate probability of class membership by means of maximization technique effectively even if we can't apply maximum likelihood estimation under incomplete conditions of random variables where we can't observe directly[5]. In text categorization it is well-known that EM algorithm works very well with a small number of (complete) training data[7].

In this investigation, we propose a method for timestamp estimation based on naive Bayes classifier via EM algorithm. We make clustering from news articles ($D_{topic,time}$,D_{time}) into temporal events. After that we assign the most suitable cluster to the news articles in D_{topic} and D_n by temporal clustering. Then we extract timestamp value and put it to a news article. By this method we can deal effectively with any topics even if they are unknown.

This work is organized as follows. In section 2 we discuss related works. In section 3 we discuss a framework of text categorization based on Bayes model by EM algorithm. Section 4 contains how we make clustering by an incremental manner and

section 5 discuss how we estimate timestamps to articles. In section 6 we show some experimental results by using TDT2 news corpus. Finally, we conclude our discussion in section 7.

2. Related Works. To our best knowledge, there have been a few relevant works so far. Among other, Mani has proposed a technique how to extract temporal aspects from documents[6]. For example, they extract index expression like "yesterday", and if timestamp of the document is April 20, the document is estimated April 19. But they put attention on relationship among temporal aspects within a document but not on inter-relationship among multiple documents nor aspects of transaction time. On the other hand, we focus on a collection of documents with both valid time and transaction time.

Papka and Yang have proposed a method for topic detection based on single pass clustering with inquery suitable for incremental situation [9, 14]. To describe documents, Papke utilizes IDF obtained from other corpus. And Yang extends IDF value incrementally. Because we suppose all data in advance, we utilize conventional TF-IDF values.

Ishikawa et al. have proposed an incremental document clustering C^2ICM based on *forgetting function*[4]. They assume probabilistic distribution and obtain similarity among documents.

Although there are some important works concerning with temporal documents as above, few work is found about timestamp estimation. Thus, for instance, we can make clustering of articles with timestamp but not without timestamps.

3. Text Categorization via EM algorithm. In this section, let us review background knowledge about text categorization by Bayes rule via EM algorithm[5]. We classify any documents $d_{time} \in D_{time}$ and $d_n \in D_n$ correctly into one of the categories by utilizing documents $D_{topic,time}$ and D_{topic} as training data sets.

EM algorithm is an iterative optimization method to estimate parameters θ of interests, given observed incomplete data $x_1, ..., x_N$. The algorithm consists of iterations of E-step and M-step. In the E-step, the algorithm estimates expected values that assign posterior probability to possible association of each individual sample. In the M-step, the algorithm tries to change θ as to maximize the likelihood of the expected values. In text categorization, EM algorithm is used very often to improve the performance of classifier learned through small labeled data by using huge unlabeled[7]. Note that the algorithm assumes all topic information are obtained only by labeled data. However, practically there must be a lot of topics not covered by the labeled data. In this investigation, we use EM algorithm for the purpose of generalizing topics and not only for the purpose of improvement performance. Then, we assign all documents to nearest generalized topics. Thus our *text categorization* means not only classifying an article into one of the known topics but also into new one.

3.1. Text Categorization by Naive Bayes. Let us review quickly about text (document) categorization by *Bayes rule*. Given a document d and categories $\mathcal{C} = \{c_1, ..., c_n\}$, we like to obtain conditional probability $p(c|d)$ which means the probability of "d belongs to $c \in \mathcal{C}$". Then we find c' such that d is expected to be in the maximum posterior category $c' \in \mathcal{C}$ where $c' = \text{argmax}_{c \in \mathcal{C}} P(c|d)$, thereby it assumes the best performance of the categorization.

To obtain $p(c_k|d)$, the likelihood and the prior probability together give joint probabilities $P(c_k)$, $P(d)$ and $P(d|c_k)$. Since $\sum_k P(c_k) = 1$ and $\sum_x P(d|c_k) = 1$, we normalize the joint probability and we have *Bayes rule* that says how *belief* should change with documents:

$$P(c_k|d) = P(c_k) \times \frac{P(d|c_k)}{P(d)}.$$

Practically we have a vast size of search space to get the probabilities, since the number of documents is really large. Accordingly, we assume every document can be decomposed into term values, denoted as a vector $d = (w_1, w_2, ..., w_m)$, w_j means a *weight* of j-th term. This *independence assumption* can be restated as follows[2]:

$$P(d|c_k) = P(d) \times \prod_{j=1}^{m} P(w_j|c_k).$$

Probabilistic categorization under this assumption is called *naive Bayes*. With this assumption, we can simplify $P(c_k|d)$ as follows:

$$P(c_k|d) = P(c_k) \times \prod_{i=1}^{|d|} P(w_j|c_k). \tag{1}$$

We use *Binary independence* approach where each document is represented by binary valued vector ($d = (w_1, w_2, ..., w_m)$, $w_j = 0$ or 1) dependent on the fact that $w_j = 1$ if and only if a document contains j-th term.

3.2. EM Algorithm. In this subsection, we discuss how to establish text categorization by Bayes rule via EM algorithm. The general algorithm consists of 4 steps:

(1) Inputs: Collections of labeled documents (D_{topic}, $D_{topic,time}$) and unlabeled documents (D_{time}, D_n).
(2) Establish an initial naive Bayes classifier $\hat{\theta}$ from the labeled documents only.
(3) Repeat until classifier parameters are saturated.
(E-step) Estimate the probability ($P(c_j|d_i; \hat{\theta})$) by using the current classifier $\hat{\theta}$. (M-step) Estimate the classifier again $\hat{\theta} = P(D|\theta)P(\theta)$ by using posteriori parameters.

(4) Output: A classifier $\hat{\theta}$.

We obtain probability $P(w_i|c_k)$ in E step by the function below:

$$P(w_i|c_k) = \frac{1 + \sum_{j=1}^{|D|} N(w_i, d_j)P(c_k|d_j)}{|V| + \sum_{i=1}^{|V|}\sum_{j=1}^{|D|} N(w_i, d_j)P(c_k|d_j)}. \tag{2}$$

D means all the documents, w_i means i-th word in the document collection and $N(w_i, d_j)$ is the count of the number of i-th word w_i in document d_j. Since we use Binary independence approach (i.e., $N(w_i, d_j) = 0$ or 1), when d_j is a labeled document, we see $P(c_k|d_j) = 0$ or 1 as given by the category label. But, if d_j is not, $P(c_k|d_j)$ is a probability given by E-step and is adjusted in M step. $P(c_j)$ is defined as follows:

$$P(c_j) = \frac{1 + \sum_{j=1}^{|D|} P(c_k|d_j)}{|C| + |D|}. \tag{3}$$

In equation(2) and (3), the values $|C|, |V|$ are introduced for *smoothing*.

The iteration process goes until saturated (within a small amount of errors).

4. Temporal Clustering. In TDT activity, it is well-known that an event corresponds to a temporal cluster[1, 14]. According to this property, we will estimate timestamp of documents accurately based on events in documents. That's why we discuss incremental clustering techniques for document stream. In this investigation, we make clustering using temporal distance between documents and clusters.

4.1. Description of Documents. We consider each news article as a set of terms (or a vector) over a collection of words extracted in advance. In fact, we have extracted only *noun* and *proper noun* words by `BrillTagger` from the articles to be learned and improved them by stemming and removing stop-words. Then we give weight to each word in the documents by *ltc* that is extension of *tf*idf* proposed by SMART system[14]:

$$w(t_j, d_i) = (1 + log_2 TF_{t_j,d_i}) \times IDF_{(t_j)}/\|\vec{d_i}\|.$$

Here TF_{t_j,d_i} means term frequency of the word t_j in the document d_i, and $IDF_{(t_j)}$ means inverse document frequency of the word t_j in the document collection. We represent a document vector $\vec{d_i}$ as follows:

$$\vec{d_i} = (w(t_1, d_i), ..., w(t_n, d_i)).$$

In the following, we represent a cluster by its centroid.

4.2. Single Pass Clustering. In this subsection, let us discuss *Single Pass Clustering* quickly by which we make incremental clustering to stream data[14]. This clustering technique provides us with a simple yet flexible technique for stream data[1].

Given a collection of clusters and a threshold value h, if a new document n has the highest similarity more than h to some cluster, the document n is appended to the cluster, and if there exists no cluster, a new cluster is generated which contains only the document n. Clearly *Single Pass Clustering* is suitable for incremental clustering to temporal data (or data stream) since, once a document is assigned to a cluster, it is not changed in the future.

The algorithm goes as follows:

(1) Let h be a threshold value.
(2) Let S be an empty set and d_1 be the first document. We generate a new cluster C_1 consisting of d_1.
(3) When a new document $d_i (i > 1)$ comes in, calculate the similarity values to all the clusters C.
(4) Let sim_{max} be the highest value and C_{d_i} the most similar cluster. If $sim_{max} > h$, add d_i to C_{d_i} and adjust the center of C_{d_i}. Otherwise, we generate a new cluster C_{d_i} that contains only d_i.
(5) Repeat the process above until no data comes.

In (4) we define $sim_{max} = MAX(sim(\vec{d_i}, \vec{C}))$. Also we define *similarity* of a document d and a cluster C where the center is V_C as below (called *cosine similarity*):

$$sim(\vec{d}, \vec{C}) = \frac{\vec{d} \cdot \vec{V_C}}{|\vec{d}||\vec{V_C}|}.$$

4.3. Forgetting Function and Time Window. In this work, we introduce a *forgetting function* [4, 14] $\omega_\lambda(t)$ where t means a lifetime and $0 \leq \lambda \leq 1.0$ for the purpose of reflecting temporal distance among documents in stream: $\omega_\lambda(t) = \lambda^t$.

When a huge amount of news articles come within a short period and we get relatively similar two documents in the collection, it is likely that we will talk about a same topic. On the other hand, even if two documents are very similar but have long temporal distance, it couldn't be.

By this function we want to separate these documents with each other. Given a cluster, the longer temporal distance a document has, the smaller value the function returns. As in figure 1, when the function returns the smaller similarity, we have the smaller clusters. Moreover, we introduce *Time Window*, and we examine similarity between clusters and documents in the window. By limiting the duration of examining, we can think about *duration* of complete forgetting, and we can cope with long estimation period. In this investigation, we define size of time window with 90 days.

[1] Actually we consider stream data as a *finite* sequence of documents.

Let us define a forgetting function as follows:

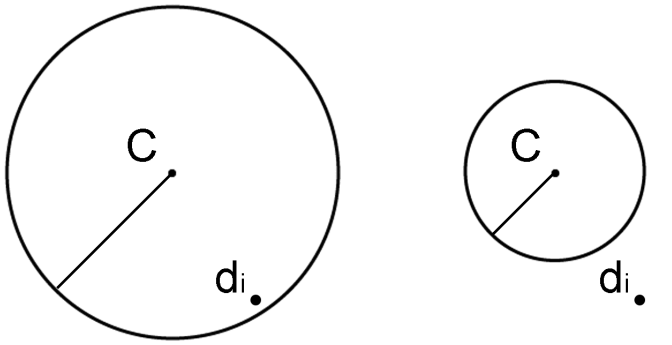
$$\omega_\lambda(t) = \lambda^t (0 \leq \lambda \leq 1.0).$$

Here t means temporal distance. [2] Then we extend a notion of similarity sim' involving the function λ as follows:

$$sim'(\vec{d_i}, \vec{C}) = \omega_\lambda(|time_{d_i} - timeC) \times sim(\vec{d_i}, \vec{C}).$$

Here $time_{d_i}$ and $time_C$ mean timestamp of a document d_i and a cluster C respectively. Then we give a timestamp $time_C$ of a cluster C as the *last timestamp* of a document in the cluster.

A figure 1 shows the current ($\omega_\lambda(0)$) and the situation 30 days after ($\omega_\lambda(30)$) where $\lambda = 0.97$. The algorithm returns a different result of clustering in accordance with temporal distance. Under this similarity we will make Simple Pass clustering.



(a)$\omega_\lambda(0) = 0$ (b)$\omega_\lambda(30) = 0.5454$

FIG. 1. *Forgetting Function.*

5. Estimating Timestamps. In this section, we discuss how to determine what event does a news article talk about. By clustering the collections $D_{topic,time}$ and d_{time}, we determine an event to an article. After that, to estimate timestamp of a news article, we utilize the result of cluster-assignment.

5.1. Cluster Assignment by k-NN. In our approach, given an article n, we select a cluster *close* to n by means of *voting* based on k-NN. Note that one topic has been assigned to all the documents $d \in D$ by Bayes rule via EM algorithm, and that each event is a subset of topics because an event is one of the temporal clusters of the topic. That's why we compare articles with each other in the topic and select the close cluster by voting. In the following, we assume *topic1* has been assigned to n.

[2] We take "day" as its unit.

When k=10, we obtain the top 10 articles similar to n from the set D_{topic1} of articles concerning *topic1*. Then we select a cluster C that accounts for the most articles. For example, in figure 2, the readers see a document n belongs to C_1.

In a case of k=1 (nearest neighbor approach) in the algorithm, we ignore the temporal clustering but uses only the result of the categorization by Bayes rule.

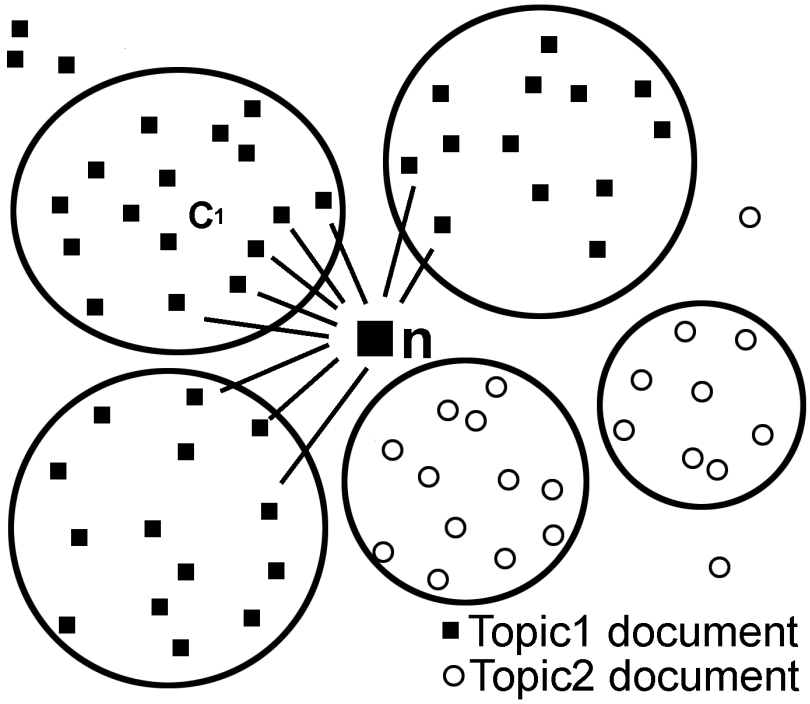


FIG. 2. *Selecting a cluster for an article* $\vec{n}$.

5.2. Timestamp Estimation. Let us estimate a timestamp of an article n by temporal clustering and the cluster-assignment above. The basic idea is that we look for the top k documents similar to n in the topic and we choose documents in the k documents which belong to the cluster C asigned to n. In figure 2, in the top 10 articles similar to n, we select 5 articles in the cluster C_1 and estimate the timestamp.

We estimate timestamp of n by examining a curve of timestamp estimation. In figure 2, let $d_{C_{11}}$ be an article in the cluster C_1. Then we give the curve of estimation from $d_{C_{11}}$ by the following rule:

$$TS_{n,d_{C_{11}},\lambda}(day) =$$
$$sim(d_{C_{11}}, n) \times distr_{C1}(time_{d_{C_{11}}}) \times \omega_\lambda(|time_{d_{C_{11}}} - day|).$$

The equation $TS_{n,d_{C_{11}},\lambda}(day)$ is illustrated as a curve of timestamp estimation in figure 3. This curve corresponds to the probability of timestamp estimation of n at day, derived from $d_{C_{11}}$, n and C_1.

Similar to clustering case, we introduce the forgetting function, then we see the probability becomes smaller in the case of longer temporal distance between $d_{C_{11}}$ and n, where, by $distr(C, time_{t_C})$, we mean a function that the number of articles at time $time_{t_C}$ obeys the distribution in a cluster C given in figure 4. We should examine the distribution because a collection of news articles arise in a short period. That's why the curve can be seen as event probability for estimating timestamp of n.

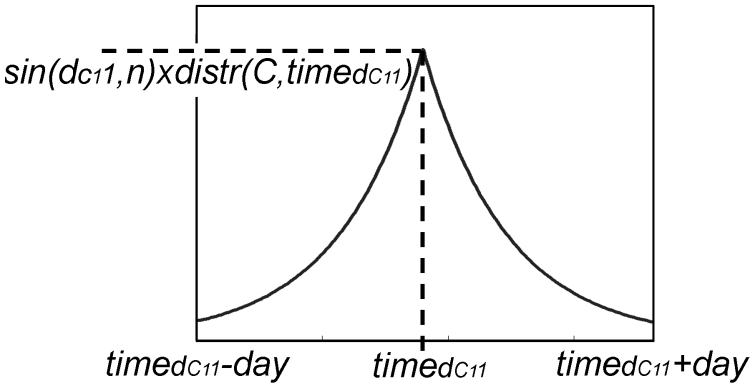


FIG. 3. *Timestamp Estimation to an article* ($\lambda = 0.97$).

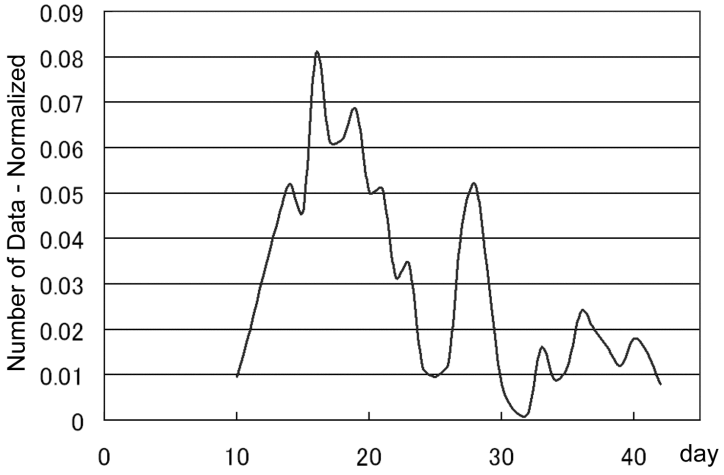


FIG. 4. *Distribution of Timestamp in a cluster.*

For each training article that is used to estimate timestamp of n, we obtain the timestamp estimation by multiplying timestamp distribution $distr(C, time_{t_{C_i}})$ and the similarity of n and t_{C_i}. Then we take all the sum to $T_{c_i} \in T_C$ of the timestamp

estimation. Eventually we get the equation below:

$$TS_{n,T_C,\lambda}(date) = \sum_{t_{ci} \in T_C} TS_{n,t_{ci},\lambda}(date).$$

For example, in figure 3, we have 5 articles in C_1. A figure 5 contains our curve of timestamp estimation.

We introduce *allowable error* of timestamp estimation which is the maximum difference between true timestamp and estimated one for the purpose of evaluation measure. In other words, this means *granularity* of timestamp estimation.

We estimate timestamp (and give it) to n by maximum likelihood principle in such a way that we maximize all the sum of the difference within the error constraint. That is, given an allowable error m days, we give some timestamp (day_n) by which we get the maximized sum of timestamp estimation before and after m days.

Formally te timestamp value day_n is given by:

$$day_n = \text{maxarg}_{day} \int_{day-m}^{day+m} TS_{n,T_C,\lambda}(day).$$

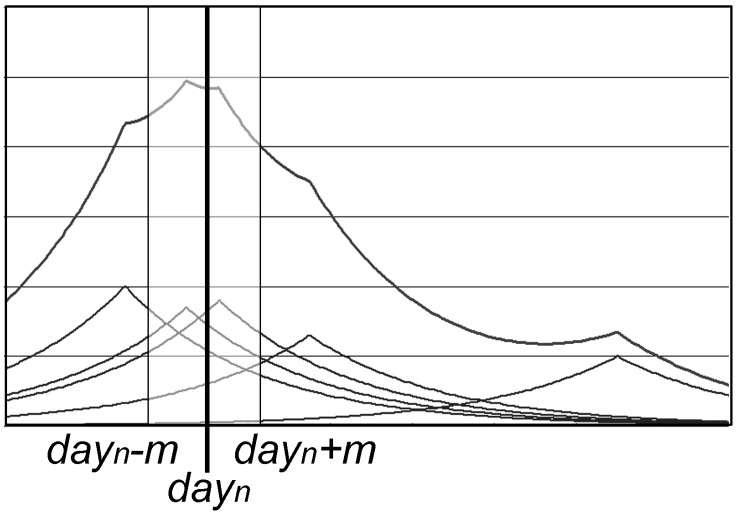


FIG. 5. *Timestamp Estimation of 5 articles.*

6. Experimental Results. To see how well our approach work, let us show some experimental results.

6.1. TDT2 Corpus. Here we examine TDT2 corpus which consists of transcribed broadcast news and newswire data during 6 months from January to June in 1998[8]. We have examined news stories in English although there are articles in Mandarin (Chinese) too. Source articles come from transcribed broadcast news

of ABC, CNN, VOA and PRI, and newswire of APW and NYT. We assume these news sources make quick reports, and there exists no significant difference among the sources about events and topics as well as the distributions.

In TDT corpus, 100 topics are defined in advance like "1998 Winter Olympics" and "Monica Lewinsky Case". To every article of each topic, one of the labels `YES, BRIEF, NO` is assigned: `YES` means the article is suitable completely for the topic and `BRIEF` partially. `NO` means none of the 100 topics is discussed. In this investigation, we utilize 8040 articles with the tag `YES` and 45580 articles with the tag `NO`. Therefore we utilize 53620 documents in total. The size of articles depends on events, it varies from less than 10 to more than 1000.

In figure 6, we show the temporal distribution of `YES` articles (described by `ON topic`), `NO` articles (by `OFF topic`) and total number of documents (by `SUM`). Off-topic articles are seen as unlabeled data to our approach. As shown in the figure, the articles are distributed uniformly but there are some bias in some topics.

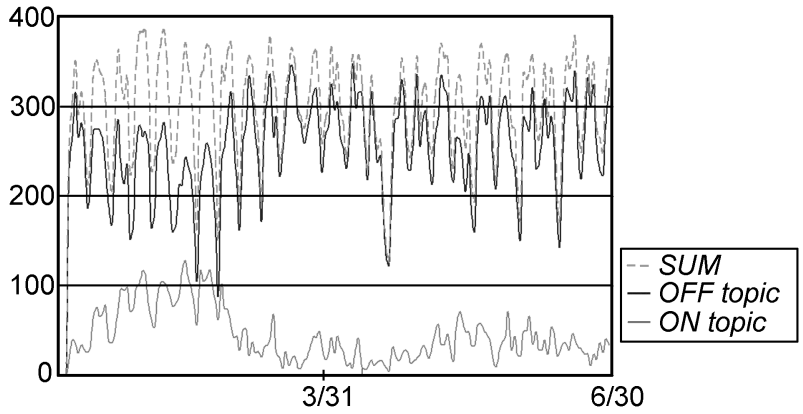


FIG. 6. *Distribution of Timestamp*

6.2. Evaluation. In this experiment, we evaluate our approach from 3 points of views.

1. Performance of timestamp estimation with incomplete data.
2. Effectiveness on the performance by EM algorithm.
3. Performance of timestamp estimation with limited topics.

First of all, we evaluate the performance of timestamp estimation with incomplete data. Since about 80% of documents have no topic information (articles with `NO`), there are a lot of topics that we can't learn from training data. We use all the TDT corpus (about 53000 documents), and divide them into two sets of training and test data as follows:

1. $D_{topic.time}$: Documents with topic information and timestamp. 1% of documents have the tag `YES`, about 80 articles.
2. D_{topic}: Documents with only topic information. 1% of documents that have the tag `YES`, about 80 articles.
3. D_{time}: Documents with only timestamp. They are 18% of TDT corpus, about 10000 articles.
4. D_u: Documents with no information about topic and timestamp. They are 80% of TDT corpus.

We select these data at random five times from $D_{topic,time}$ and D_{topic}, we estimate timestamp articles in D_{topic} and D_u, and finally we evaluate the performance.

In this experiment, we give 3 cases of allowable errors: 1week (7days), 2weeks (14days) and 1month (30days). When we give an allowable error "1week", we say timestamp is *correctly estimated* if the difference between correct timestamp and estimated timestamp is less than 1week. Moreover, we will examine 5 kinds of k (k=1,2,5,10,30) in k-NN and Nearest Neighbor (NN). Note that NN is different from 1-NN: in NN, we select the nearest document, while 1-NN means we select the nearest document *in the topic* assigned (through the categorization).

Next we examine how EM algorithm works to timestamp estimation under several numbers of iteration: 0(Naive Bayes), 5, 10 and 20 times.

Finally we evaluate the performance of our approach with limited topics. We utilize `YES` articles only (about 8000 articles)[12]. We have divided these data just same as the previous case. And then we will compare the estimation using EM algorithm to the one without EM algorithm.

6.3. Results. First of all, let us show our results of timestamp estimation with incomplete data in table 1 and figure 7.

TABLE 1
Timestamp accuracy with incomplete situation (1).

(EM=10)

(%)	1week	2weeks	1month
NN	31.80	40.64	54.55
k=1	36.83	45.40	58.46
k=3	36.98	46.05	60.39
k=5	35.17	44.57	59.50
k=10	32.33	42.33	58.31
k=20	14.85	30.37	57.88

A table 2 and a figure 8 contain the effectiveness on the performance by EM algorithm.

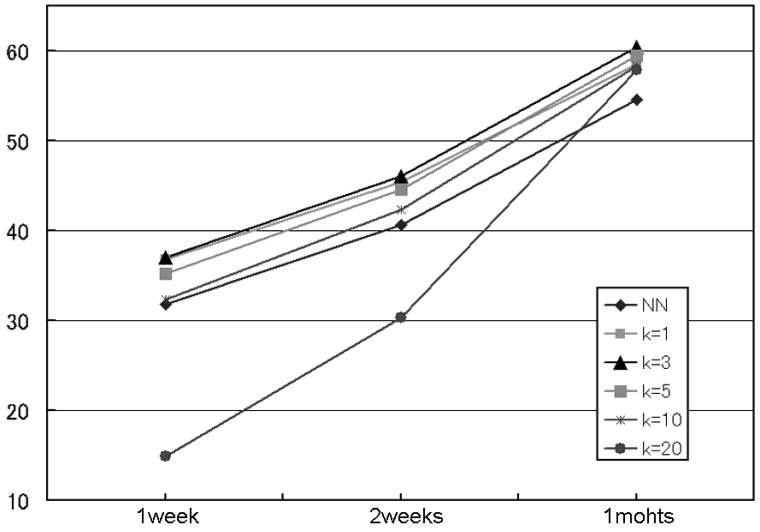


FIG. 7. *Timestamp accuracy with incomplete situation (2).*

TABLE 2
Effectiveness by iteration of EM algorithm (1).

(K=3)

(%)	1week	2weeks	1month
Non-EM	35.33	43.97	57.90
EM5	36.07	44.90	58.92
EM10	36.98	46.05	60.39
EM20	36.07	44.84	58.74

Finally we show the result of timestamp estimation with limited topics in table 3 and figure 9.

TABLE 3
Timestamp accuracy with limited topics (1).

(K=5,EM=10)

	1week	2weeks	1month
OnTopic+EM	56.75	69.17	81.61
OnTopic(NonEM)	50.05	60.87	73.55

6.4. Discussion. As illustrated in table 1 and figure 7, we got high quality of timestamp accuracy in the incomplete situation: about 37% with allowable error 1week, and about 60% with allowable error 1month.

Compared to NN case, all of k-NN are superior for timestamp estimation. There-

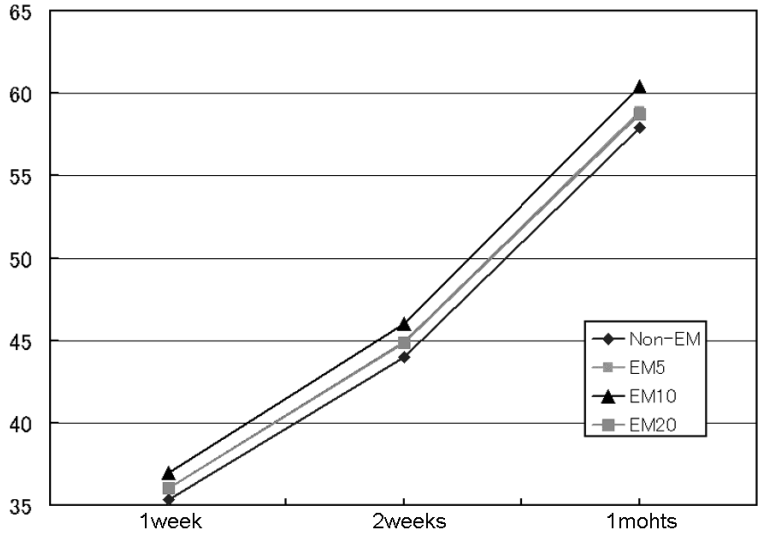


FIG. 8. *Effectiveness by Iteration of EM algorithm (2).*

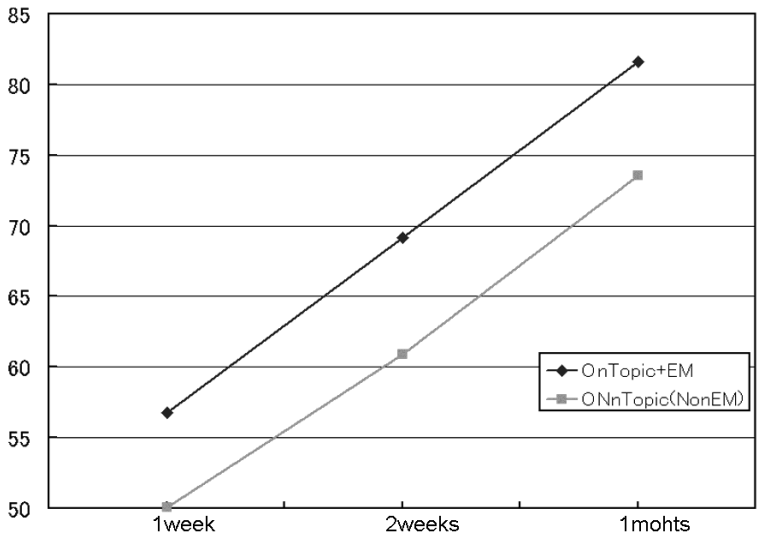


FIG. 9. *Timestamp accuracy with limited topics (2).*

fore we can say that our approach, timestamp estimation by using topic and event information, is suitable for timestamp estimation.

In table 2 and figure 8, we have shown the effectiveness on the performance by EM algorithm. In a case of 10 iterations, we got the best accuracy, but in a case of 20 times, we got the worst accuracy. In text categorization, topic information is assigned by hands. For this reason, word frequency doesn't correspond to distribution of topics. EM algorithm may increase small erroneous states given initially but it may

not always improve categorization performance. Some investigation looks for optimal iteration of EM algorithm[10].

As shown in table 3and figure 9, we got good results of timestamp estimation with limited topics. In fact, we got about 60% with allowable error 1 week, and about 83% with allowable error 1month. In both cases, we see timestamp estimation by EM is superior than the one by naive Bayes. We can say EM algorithm is suitable for timestamp estimation, and we can cope with topics not defined in advance.

In this experiment, we got the best result with threshold $th = 0.1$, and a forgetting function of $\lambda = 0.97$.

In table 4, we see the timestamp estimation performance to each period. Generally the performance have been improved by EM algorithm, but in the first month, the performance becomes worse. This is because there exists no article before and the algorithm takes the amount of topics/events into consideration. It is worth to note that we got good performance at the period without enough amount of data. In short, we can estimate timestamp well to incomplete situation by EM algorithm.

TABLE 4
Timestamp accuracy of each period.

(K=5,EM=10,Arrowable 2weeks)

(%)	Jan	feb	Mar	Apr	May	Jun
EM	34.40	50.53	43.63	40.94	43.11	48.42
NonEM	39.10	48.92	41.44	39.37	40.90	44.21

7. Conclusion. In this investigation, we have discussed how to estimate timestamp of news articles in stream with incomplete situation. In our experiments we got high quality of timestamp accuracy in the incomplete situation with k-NN technique, about 37% with allowable error 1week, and about 60% with allowable error 1month. Also we got good results of timestamp estimation with limited topics, about 60% with allowable error 1 week, and about 83% with allowable error 1month. We have shown EM approach works very well in this case. All these show our approach is promising.

There remain several works. One of the issues is we need to distinguish "incomplete" articles from "outlier" ones. Also we are targeting for other kinds of collection like Web pages and Technical journals.

Acknowledgment. We would like to thank Prof.Wai Lam in The Chinese University of Hong Kong for his helpful comments and encouragement. Also we would like to acknowledge the financial support by Grant-in-Aid for Scientific Research (C) (No.16500070) from Ministry of Education, Culture, Sports, Science and Technology in Japan.

REFERENCES

[1] Allan, J., Carbonell,J., Doddington,G., Yamron,J. and Yang,Y., *Topic Detection and Tracking Pilot Study: Final Report*, Proc. DARPA Broadcast News Transcription and Understanding Workshop (1998).

[2] Lewis.D. D., *Naive (Bayes) at forty; The independence assumption in information retrieval*, in: Proceedings of ECML-98, 10th European Conference on Machine Learning, 1998.

[3] Grossman, D. and Frieder, O., *Information Retrieval - Algorithms and Heuristics*, Kluwer Academic Press,1998

[4] Ishikawa, Y. and Kitagawa, H., An Improved Approach to the Clustering Method Based on Forgetting Factors, in proc. 5th European Conference on Research and Advanced Technology for Digital Libraries (ECDL'01),2001

[5] Iwasaki, M., Statistic Analysis for Incomplete Data, EconomistSha, Inc., 2002 (in Japanese)

[6] Mani,I. Wilson, G., Robust temporal processing of news. In Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL 2000), pages 69–76, New Brunswick, New Jersey, 2000. Association for Computational Linguistics.

[7] Nigam,K. Mccallum,A. Thrun,I. and Mitchell, T., Text Classification from Labeled and Unlabeled Documents using EM, Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

[8] *National Institute of Standards and Technology(NIST)*,http://www.nist.gov/speech/tests/tdt/

[9] Papka,R. and Allan, J., *On-line new event detection using single-pass clustering*, Technical Report UMASS Computer Science Technical Report 98-21, Department of Computer Science, University of Massachusetts, 1998.

[10] Shinnou,H. and Sasaki, M., *Unsupervised Learning of Word Sense Disambiguation Rules by Estimating an Optimum Iteration Number in EM Algorithm*, IPSJ Journal, Vol.44, No.12, 2003(in Japanese).

[11] Uejima,H., Miura,T. and Shioya,I., *Improving Text Categorization by Synonym and Polysemy*, IEICE Trans.on Info. & System J87-D-I-2, pp.137-144, 2004 (in Japanese).

[12] Uejima,H., Miura,T. and Shioya,I., *Giving Temporal Order to News Corpus*, The 16th IEEE International Conference on Tools with Artificial Intelligence, 2004.

[13] Wayne,C., Doddington,G. et al., *TDT2 Maltilanguage Text Version 4.0 LDC2001T57*, Philadelphia:Linguistic Data Consortium (LDC), 2001.

[14] Yang,Y., Pierce,T. and Carbonell,J., *A Study on Retrospective and On-Line Event Detection*, Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval.