Using Contextual Analysis for News Event Detection

W. Lam,* H. M. L. Meng, K. L. Wong, J. C. H. Yen
Department of Systems Engineering and Engineering Management,
The Chinese University of Hong Kong, Shatin, Hong Kong

The rapidly growing amount of newswire stories stored in electronic devices raises new challenges for information retrieval technology. Traditional query-driven retrieval is not suitable for generic queries. It is desirable to have an intelligent system to automatically locate topically related events or topics in a continuous stream of newswire stories. This is the goal of automatic event detection. We propose a new approach to performing event detection from multilingual newswire stories. Unlike traditional methods which employ simple keyword matching, our method makes use of concept terms and named entities such as person, location, and organization names. Concept terms of a story are derived from statistical context analysis between sentences in the news story and stories in the concept database. We have conducted a set of experiments to study the effectiveness of our approach. The results show that the performance of detection using concept terms together with story keywords is better than traditional methods which only use keyword representation. © 2001 John Wiley & Sons, Inc.

1. INTRODUCTION

Newswire stories are created and stored electronically everyday at many news agencies. Users may receive news streams from multiple sources. Traditional query-driven retrieval is useful when you precisely know the nature of the events or facts you are seeking. Generic queries such as "What happened?" or "What's new?" are not suitable. Browsing in large-scale information spaces without guidance is not effective. Suppose, for example, a person who has returned from a long vacation and wants to find out what happened during the period. It is impossible to read the whole news collection and it is unrealistic to generate specific queries about unknown facts. As a result, it is difficult to retrieve or to check all the potentially relevant stories.

Clearly, it is beneficial to have an intelligent agent to automatically locate topically related stories in a continuous stream of news articles. The problem becomes even more challenging if the news sources are multilingual. The topic

*Author to whom correspondence should be addressed; e-mail: wlam@se.cuhk.edu.hk.

detection and tracking (TDT) project, sponsored by the Defense Advanced Research Projects Agency (DARPA) in the United States, is also investigating this area.[1] *Event or topic detection* is one of the tasks in the TDT project.

2. PROBLEM DEFINITION

The objective of event detection is to detect topically related stories from a continuous stream of news. An *event* is defined to be a seminal event or an activity, along with all directly related events and activities and their supporting discussions. For example, a story on the outbreak of bird flu or its treatment, will be considered to be an event. Events are not predefined in the system and they are discovered online during the processing of news stories. Figure 1 illustrates the detection system process. A stream of stories written in different languages is arriving in chronological order. The system should be able to identify each story whether it is discussing a new event that has not been discussed by earlier stories or is discussing an event that has been previously detected.

The detection technology can be employed in different applications. For example, it can be used for generating a temporal evolution of different events. It is also useful for producing structured guidelines for story navigation of the whole news collection. Finally, it can be used for analyzing the content shifts of a particular event.

Our corpus consists of English and Chinese news. One issue for event detection is to deal with multilingual news content. In this research, multilingual news actually refer to English and Chinese news. English and Chinese are quite different in grammatical style and structure. Determining whether an English story and a Chinese story report the same event becomes a great challenge in multilingual settings.

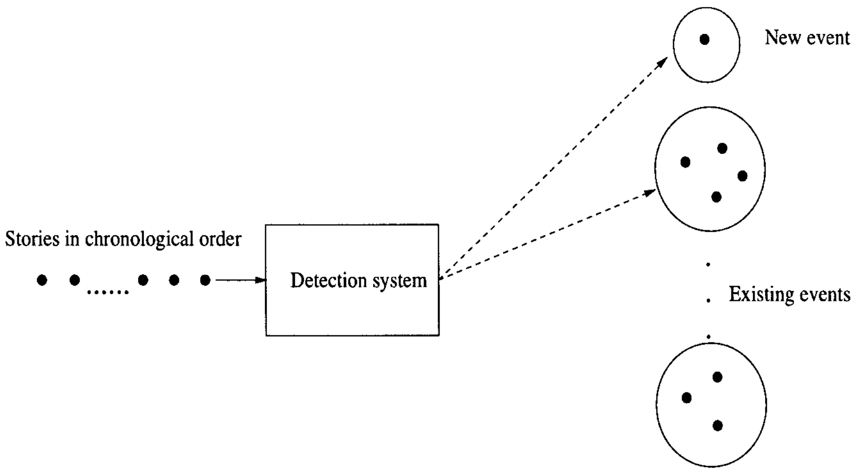


Figure 1. Event detection process.

We develop a new method which does not solely rely on the simple key terms that appeared in the story. Our method also considers the *concept terms* and *named entities* such as person, location, and organization names. Concept terms of a story are derived from statistical context analysis on a separate concept database. Specifically, they are extracted from the concept database according to the relevance between stories in the concept database and sentences in the news story.

3. RELATED WORK

There are some related works in this research area. The Dragon Systems used unigram distributions for document representation and applied k-means clustering for grouping related documents.[1,17] The first story in the corpus represents the seed of an initial cluster. The subsequent stories are compared to the existing clusters one by one. A story is inserted to the most similar cluster or formed as a seed of a new cluster. Carnegie Mellon University used the combined agglomerative and single-pass clustering systems called BORG.det with different term-weighting schemes term frequency−inverse document frequency (TF−IDF) and language modeling to perform event detection.[18,19] One of the systems is a cosine-similarity clustering engine called GACIncr. Another system is called Incr.LM. It uses language models trained with expectation maximization (EM) on member documents for clustering representation. The University of Massachusetts proposed a detection system which supports two kinds of clustering algorithm.[2,3] One is agglomerative centroid clustering and another is k nearest neighbor method.

Most of the existing methods use simple keywords to represent news articles. Hence, the similarity measure used in the clustering algorithm is based on the keyword representation. However, if there are two documents describing similar events but using different styles and words, the system has difficulty in making a correct decision that they belong to the same event. This is the problem of vocabulary switching, in which traditional text processing based on keyword comparison could not provide good performance.[5]

4. CONCEPT TERM MODEL

4.1. Background of Contextual Analysis

The purpose of concept term generation is to determine appropriate concept terms for a news story. A major resource is a separate concept database. The concept database contains potential concept terms derived from a concept generation corpus. The concept generation corpus is actually a separate collection of news stories, called *concept generation news*. Potential concept terms are key terms contained in the stories. In our experiments, the corpus consists of two collections, namely, Associated Press (AP) 1990 and Wen Wei Po. The AP corpus contains 78,304 English news and the Wen Wei Po corpus contains 6127 Chinese news. When the detection system receives an incoming story, these potential concept terms will be evaluated according to the degree of relevance between the incoming story and the concept generation news. Concept terms with high relevance will be selected to represent the story. This concept representation of a news story will be involved in computing the similarity during the detection process. We will describe in detail the idea of concept term generation in the next subsection.

Concept terms can be used for dealing with the problem of vocabulary switching.[15] The idea of using concepts has been applied for query expansion.[16] It combines the technique of global analysis and local feedback between a query and documents. Concepts are selected based on the co-occurrence with the query terms. This idea has been extended and used for text segmentation.[12] Using cluster analysis to generate concept spaces has been applied to support the searching of scientific literature, where two biological fields used two different sets of vocabularies to describe the same thing.[7] A combination of natural language parsers, concept space, and category maps have been applied to overcome the vocabulary switching problem in medical information retrieval.[9] To link stories that use different sets of vocabularies together, the approach of concept association has been applied to improve the quality of searching.[6] Evaluation on the performance of using category map and concept space techniques in supporting Internet browsing and searching has been conducted and results indicated that the precision has improved.[4]

4.2. Concept Term Generation

4.2.1. Concept Generation Algorithm

The concept database stores all the potential concept terms derived from the concept generation news. Potential concept terms are basically keywords from the concept generation news. As mentioned above, the concept generation corpus used in our experiments consists of two collections, namely, Associated Press (AP) 1990 containing English news and Wen Wei Po containing Chinese news. Words in English AP stories are stemmed and stop words are filtered out. Chinese stories in Wen Wei Po are segmented into meaning words. These words, in their native language, form the basis for potential concept terms. In particular, potential concept terms are extracted and associated with the concept generation stories. These extracted terms form two concept databases, one for English and one for Chinese.

When an incoming news story, D, arrives for detection, we break it into sentences for subsequent processing. The concept term determination for this story is conducted via the following steps:

(1) Each sentence in the current story, D, is posed as a query Q. The concept generation news of the corresponding language will be compared with the query Q and the terms of top n relevant stories will be retrieved from the concept database. We use a vector space model to determine the relevance. Thus, each sentence in D and the stories in the concept database are transformed to vector representation for comparison. Table I shows an example of a query and its most relevant story is shown in Table II.

Table I. An example of a query to the concept database.

莫斯科时间今晨零时十三分俄罗斯宇航员索洛维耶夫和美国宇航员沃尔夫进行了这个乘员组第7次,也是最后一次太空行走

Table II. The most relevant story to the query in Table I.

周二，俄罗斯技术人员在莫斯科东南面‘星’研究中心检查宇航员的太空服。俄罗斯宇航员将於明年穿这件太空服与美国宇航员一起在‘和平’号太空站工作

(2) The potential concept terms are the terms in the top n relevant stories in the concept database. Table III shows the potential concept terms for the story in Table II. A score is computed for each concept term to reflect the degree of relevance to the sentence. The concept frequently co-occurring with the query terms and infrequently occurring in the database will get a higher score. Consider a concept c, the score is calculated as

$$r(Q,c) = \prod_{t_i \in Q} (\delta + A(c,t_i) idf_c) \tag{1}$$

where $idf_c = \log(N/N_c)$, $A(c,t_i) = \log(\sum_{j=1}^{j=n} f_{t_{ij}} f_{c_j})$, $f_{t_{ij}}$ is the number of occurrences of t_i in story $j(s_j)$, f_{c_j} is the number of occurrences of c in s_j, N is the number of stories in the collection, N_c is the number of stories containing c, δ is 1 for avoiding the score of the concept to be zero.

(3) Concept terms for each sentence will be weighed according to the location of the query sentence in the story. In most news stories, the first few lines are the summary of each story. They are more important than the remaining part of the story. A weighting scheme is, therefore, applied to deal with this situation. The weighting of each concept term is related to the location of the sentence from which the concept is associated. For example, concept terms retrieved by the first sentence will have a higher weight than the concept terms from other sentences. The weighting formula is a simple linear function shown as

$$W(Q,c) = \frac{(N_s - 1) + (L_b - 1)(s - 1)}{(N_s - 1)} \tag{2}$$

Table III. Potential concept terms for Table II.

周二	和平
东南	检查
明年	传真
美联社	工作
研究中心	美国
俄罗斯	一起
技术人员	宇航
莫斯科	太空

where N_s is the number of sentences in this story, s is the sth sentence in the story retrieved by the concept c, L_b is the weighting of the concepts retrieved by the last sentence and it should be set in $[0, 1]$. Therefore, the $W(Q, c)$ should be bounded by $[L_b, 1]$. Note that $W(Q, c)$ of the concept terms retrieved by the first sentence is 1 whereas the weight of the concept terms retrieved by the last sentence is L_b.

(4) After all sentences have been processed for the current story D, the total score of each concept term to D can be computed as

$$T(D,c) = \sum_{Q \in D} (r(Q,c) * W(Q,c)) \tag{3}$$

Top concept terms with the highest total scores are used to represent the story. Table IV shows an example of an original story and Table V shows the highest concept terms for this story.

From the example, we discover that some concept terms like "云南省" do not appear in the original story but are very related to the story content. The concept terms returned from the concept database are strongly associated with the original sentences since the method favors the co-occurrence with multiple query terms in the sentence. Those concept terms can provide extra information for the story.

Table IV. An example of an original Chinese story.

为丰富群众节日生活,昆明市城区和各郊县举办了丰富多彩的迎新年体育活动。昆明市体委、教委等部门今天上午组织五华、盘龙两城区大中小学学生元旦越野赛跑 ,近3000名学生分成年、中学、小学组参加了男子10公里、女子5公里越野赛跑

Table V. Concept terms generated for the story in Table IV.

昆明	1260.181763	云南省	294.445190
越野	277.965240	西南	250.608826
运动会	217.269150	云南	146.948563
接壤	92.794609	宣传部	63.658455
天然	45.024376	经贸	42.977352
潜力	39.672085	田赛	36.468018
少年	35.845299	揭幕	35.283073
游泳	33.227982	贸易	26.160017
市政局	25.133614	机构	22.434792
年华	21.695694	条件	20.561026
比赛	20.439396		

4.2.2. Concept Term Representation for Detection

After processing all the stories, each story is associated with a number of top relevant concept terms and their weights. These concept terms become a part of story information for event detection. Similar to traditional keyword representation of story contents, concept terms and their weights are recorded for each story.

5. EVENT DETECTION MODEL

In general, our proposed system consists of several modules, as shown in Figure 2.

5.1. Story Representation

We employ a modified vector-space model to represent a story. Traditionally, every story is represented by a vector of weighted terms. In our approach,

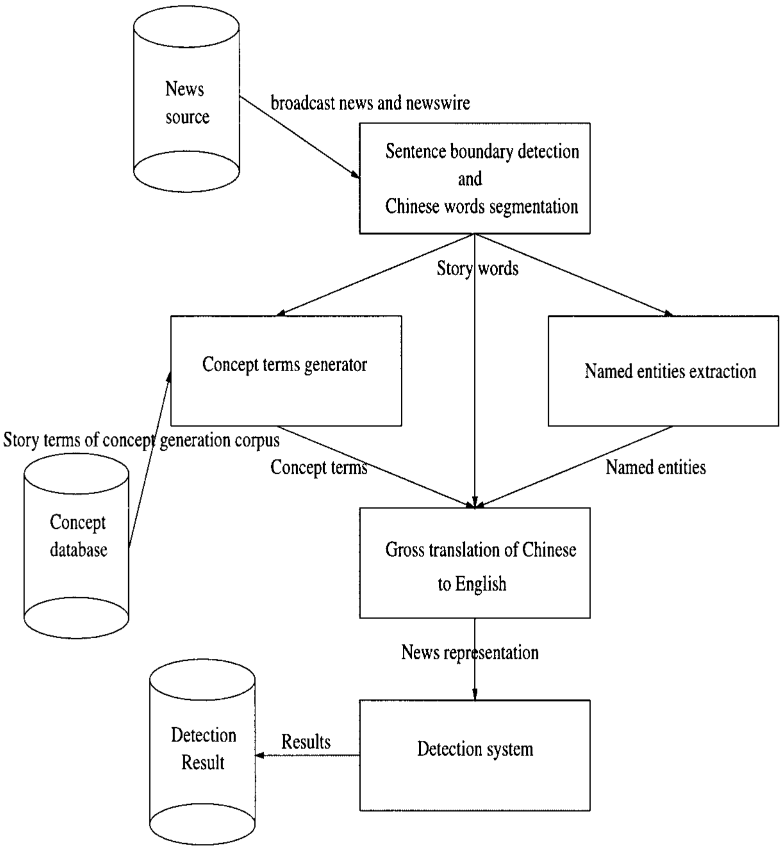

News source
broadcast news and newswire
Sentence boundary detection and Chinese words segmentation
Story words
Concept terms generator
Named entities extraction
Story terms of concept generation corpus
Concept database
Concept terms
Named entities
Gross translation of Chinese to English
News representation
Detection Result
Results
Detection system

Figure 2. Block diagram of our proposed approach.

we extract three kinds of elements from a story. Each story is represented by three kinds of vectors, namely, the concept feature vector, the named entity feature vector, and the story feature vector. Whenever a story is received, it is converted into such representation. The terms contained in these vectors are in the same native language as the original story, i.e., English for English stories and Chinese for Chinese stories. In vector representation, stories can be regarded as instances in multidimensional space. The similarity between two stories, called similarity score, captures the closeness of the content of the stories. Basically, this score can be calculated based on the term weights of these three vectors found in the story representation.

For each story S, its concept representation is expressed by a concept feature vector as

$$\{w(S,c_1),w(S,c_2),\ldots,w(S,c_m)\} \tag{4}$$

where each component $w(S,c_i)$ represents the weight of the corresponding concept term c_i among the top m concept terms extracted.

In traditional story vector representation, the weightings are related to term frequencies and inverse document frequencies. In our concept feature vector, however, those weightings are the scores of the concept terms in the story. This concept score measures the degree of relevance of the concept to the story. The higher the score, the more important the concept term is.

The second element of story representation is the named entity feature vector. This vector contains named entities of a story. The named entities are derived from a transformation-based error-driven part-of-speech tagger.[11] The named entity feature vector for a story S is represented as

$$\{w(S,e_1),w(S,e_2),\ldots,w(S,e_n)\} \tag{5}$$

where each component $w(S,e_i)$ represents the weight of the corresponding named entity e_i. The higher the weight, the more important the named entity is. Table VI shows an example of a partial story and Table VII shows the named entities and their corresponding weights in the named entity feature vector. In this example, the terms enclosed by ⟨ORG⟩ represent organization terms in the story. Similarly, the terms enclosed by ⟨PLACE⟩ represent the location term and the terms enclosed by ⟨PERSON⟩ represent person names in the story.

The weights of the terms in named entity feature vectors are determined by the traditional TF−IDF scheme. The "TF" is the term frequency while "IDF"

Table VI. An example of an English story.

A subdued Hong Kong on Wednesday marked the first anniversary of its return to China, confident that the mainland would let it manage its own affairs but increasingly uncertain of its ability to do so. A harbor-side flag-raising ceremony on the site of last year's handover ceremony and an all-star variety show offered Hong Kong's 6.5 million people a colorful spectacle. Chinese President Jiang Zemin was to attend an evening banquet in honor of the anniversary.

Table VII. Top ranked named entities for the story in Table VI.

⟨ORG⟩	
European Union	3.219394
Portuguese	2.439383
⟨/ORG⟩	
⟨PLACE⟩	
Hong Kong	33.612464
China	15.052707
Beijing	5.819403
Macau	6.440619
Taiwan	4.330921
⟨/PLACE⟩	
⟨PERSON⟩	
Jiang Zemin	2.626682
Jiang	8.333850
Clinton	2.506171
Tung	5.805778
Chee-hwa	2.982070
Martin Lee	3.489155
Lee	5.618416
⟨/PERSON⟩	

represents the inverse document frequency. This formulation enhances the term which appears in fewer stories while it penalizes the term which appears in many stories. The weight of a named entity e in story S is calculated by

$$w(S, e) = tf_e * idf_e \tag{6}$$

where $idf_e = \log_{10}(N/df_e)$ is the inverse document frequency of the term e, N is the number of stories currently processed, df_e is the number of stories among the N stories that contain the term e, tf_e is the number of occurrences of e in the current story. As we can see, the named entity can get a higher weight if it frequently occurs in the current story and if it infrequently occurs in the past stories. The top ranked named entities with corresponding weights are used to form the named entity feature vector of the story.

The third element of story representation is the story feature vector. A story feature vector, similar to a named entity feature vector, is represented as

$$\{w(S, t_1), w(S, t_2), \ldots, w(S, t_k)\} \tag{7}$$

where each component $w(S, t_i)$ represents the weight of the term t_i among the k terms extracted. These weights represent the importance of the corresponding terms to the story. The higher the weight, the more important the term to the story. Table VIII shows the story terms and their corresponding weights in the story feature vector for the story in Table VI. For English stories, each term is converted to its word stem by a stemming algorithm.[13]

Table VIII. Top ranked story terms in Table VI.

Handov	0.309129	Econom	0.295747
Territ	0.289123	Confid	0.245764
Year	0.232134	Freedom	0.226972
Annivers	0.216274		

The calculation of the weights in a story feature vector is similar to that in a named entity feature vector except that the location of the story term in the story is also considered. The use of the location of the story term in calculating the story term weight is to handle the fact that the first few lines are usually more important in most news stories. For instance, the first few paragraphs usually contain the summary of the news content. The subsequent paragraphs give the details and additional information of the news. We give higher weights for the terms located in the earlier sentences of the story and we give lower weights for the terms in the remaining sentences. Calculating the sentence weighting of the term t in sentence E is similar to that in concept term generation. The formula is as

$$L(E,t) = \frac{(N_s - 1) + (L_b - 1)(s - 1)}{(N_s - 1)} \tag{8}$$

where N_s is the number of sentences in this story, s is the sth sentence in the story retrieved by the story term t, L_b is the sentence weight of the story terms extracted from the last sentence and it should be set in [0, 1]. Therefore, the sentence weighting should be bounded by $[L_b, 1]$, in which the weighting of story terms extracted from the first sentence is 1 whereas the weighting of story terms extracted by the last sentence is L_b.

The weight of the story term t in each sentence E is computed as

$$e(E,t) = idf_t * L(E,t) \tag{9}$$

where idf_t is the inverse document frequency of t. The total weight of the story term t in the story S is the summation of all the weights of that term in each sentence of the story:

$$w(S,t) = \frac{\sum_{E \in S} e(E,t)}{\max_{t_i \in S} w(S,t_i)} \tag{10}$$

The denominator of Eq. 10 is the normalization factor, which is the maximum value of all the story term weights in the story. Top ranked story terms are selected with the corresponding weights to form the story feature vector of the story.

5.2. Event Representation

In our approach, an event is represented in a similar way as a story. Specifically, an event is represented by three vectors, namely, the concept feature vector, the named entity feature vector, and the story feature vector.

The dimension of the event vectors can be expanded when a new term from a relevant story is inserted. The weight of an existing term can be updated by computing the mean of the existing term weight in the event with the term weight in the story vector. For example, assume that the story feature vector of event C contains four terms with corresponding term weights: $\{(t_1, 1.0), (t_2, 0.8), (t_3, 0.3), (t_4, 0.1)\}$. A story with the story feature vector $\{(t_3, 1.0), (t_1, 0.7), (t_5, 0.2)\}$ is to be assigned to the event. Then, the updated story feature vector of the event is the mean of the two vectors: $\{(t_1, 0.85), (t_3, 0.65), (t_2, 0.4), (t_5, 0.1), (t_4, 0.05)\}$.

5.3. Similarity Score

With the concept feature vector, the named entity feature vector, and the story feature vector defined for stories and events, we can estimate the similarity score between a story and another story, as well as a story and an event through a similarity function. This similarity function basically consists of three scores, namely, the concept relevance score, the named entity relevance score, and the story feature relevance score. In addition, we introduce a time-adjustment scheme when calculating event-related similarity. We use the time-adjustment scheme to reflect the fact that stories that happened far apart in time unlikely belong to the same event.

In our approach, stories and events are represented by vectors. Given vector $V_i = \{w(S_i, t_1), \ldots, w(S_i, t_n)\}$ and vector $V_j = \{w(S_j, t_1), \ldots, w(S_j, t_m)\}$, the similarity score is the cosine of the angle between the vectors, called the angular separation,

$$s_{ij} = \frac{\sum_k^m \sum_l^n a_{kl}}{\sqrt{\sum_{k=1}^n w(S_i, t_k)^2 \sum_{k=1}^m w(S_j, t_k)^2}} \tag{11}$$

where

$$a_{kl} = \begin{cases} w(S_i, t_k) * w(S_j, t_l) & \text{when } t_k = t_l, \\ 0 & \text{when } t_k \neq t_l. \end{cases}$$

The similarity score s_{ij} is in the range of $[0, 1]$. When s_{ij} is 0, it means that the two vectors are totally dissimilar. When s_{ij} is 1, it means that the two vectors are totally equal. Three scores are first computed from corresponding vectors by using Eq. 11. For example, if we want to compute the distance between two stories, the story feature relevance score S_s is computed by comparing the story feature vectors of the stories. The named entity relevance score S_n is computed by comparing the named entity feature vectors of the stories. The concept relevance score S_c is computed by comparing the concept feature vectors of the stories. The final similarity score S_f between the two stories is the summation of these three scores weighted by the corresponding parameters as shown below:

$$S_f = S_s * w_s + S_c * w_c + S_n * (1 - w_s - w_c) \tag{12}$$

The parameters w_s, w_c are in the range of [0, 1]. These parameters control the degree of emphasis on the corresponding vectors. For example, if we want to put more emphasis on story terms and less emphasis on concept terms, we set w_s higher and we set w_c lower.

Our proposed method based on three vectors instead of a traditional single vector for the story representation allows us to use three different kinds of features to represent the story. Story terms represent the general content of the story. Concept terms represent the concept related to the story. Named entities represent some specialized items of the story.

5.4. Gross Translation Method

After a story is represented by vectors, we conduct gross translation for Chinese terms into English terms and we generate new vectors. Here, full-fledged translation is not needed since our purpose is to translate the terms suitable for conducting the event detection. A bilingual dictionary and a pin-yin file are used when necessary to perform translation.[10] The bilingual dictionary contains English translations for each Chinese word as shown in Table IX. The pin-yin file contains Mandarin pin-yin for each Chinese character as shown in Table X. The translation algorithm actually is a simple search and match procedure described below:

(1) Look up each Chinese term in the bilingual dictionary.
(2) If there exists corresponding English words in the dictionary.
    (a) Get the corresponding English word sets.
    (b) Filter out the stop words.
    (c) Stem each English term.

Table IX. A bilingual dictionary.

经营	/engage in/run/operate/
经营费用	/business cost/business expense/
经营管理和维护	/Operations Administration and Maintenance/OAM/
经由	/via/
井	/warn/well/
井里汶	/Cirebon/
警	/to alert/to warn/
警报	/alarm/alert signal/alarm/alert/warning/

Table X. A pin-yin le.

百	bo2
摆	bai3
佰	bai3
败	bai4
拜	bai4
稗	bai4
斑	ban1

(3) Else if the Chinese term represents the name of a person, get the pin-yin for it. For example, “江泽民” will become “Jiang Zemin”.
(4) Else, throw it away.

As a result, each Chinese story is now represented by a set of English terms contained in three kinds of vectors.

5.5. The Detection System

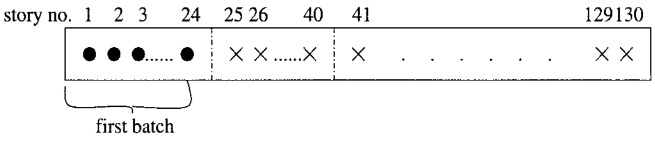
5.5.1. *Detection Requirement*

The event detection task sequentially processes stories from different sources in chronological order. When the event detection system receives a batch of stories from a source, it needs to decide, for each story, whether it is related to some existing events or discusses a new event. The event detection system is allowed to decide the events until some amount of subsequent batches of stories are read into the system. This amount is controlled by a parameter called the *deferral period* which is the number of batches of stories for which the system sees before the event identification decisions need to be made.

5.5.2. *The Top Level Model*

In general, our detection system consists of three main components, namely, similarity calculation, grouping related elements, and event identification. These three components are discussed in detail in later sections. At the beginning of the detection task, the system will read a number of batches of stories up to the deferral period. The stories are stored in a story list. Figure 3 shows an example of 130 stories loaded into the story list. The first group of stories, i.e., stories 1−24 in this example, come from a single batch.

The next step is to conduct clustering on the stories in the story list. The stories in the first batch in the story list are examined and their events are extracted according to the clusters to which they belong. After the events of



● : Stories in the first batch

× : Stories in the remaining list

Figure 3. An example of a story list.

those stories in the first batch are obtained as output, these stories are cleaned up and new stories are loaded from the next incoming batch. Then, the second batch of stories in the story list becomes the first batch. The above procedures will be repeatedly proceeded until no more batches exist. As a result, the system will output the detection result for each story with the corresponding event and a confidence score.

A major component of our event detection approach is the clustering method. In general, clustering is a technique to find appropriate groupings for a set of data.[8,14] The data in each cluster should be similar to one another while the data in different clusters should be dissimilar. We employ a modified agglomerative clustering as the core algorithm. Agglomerative clustering is a hierarchical clustering algorithm. At the beginning, each data is regarded as a singleton cluster containing one element. Then, we compute all pairwise similarity between clusters. The clusters are then merged based on the similarity between the clusters. There are different methods to find the distance between two clusters. For instance, in *group average clustering*, the distance between two clusters is defined as the average of the distances between pairs of data in the clusters. In *centroid clustering*, the distance between two clusters is defined as the distance between the centroid of the two clusters.

In our modified agglomerative clustering approach, we adopt the centroid clustering since the number of stories in a cluster could be very large. We introduce a kind of clusters, called temporary clusters, in the clustering process to handle the incremental nature of the detection problem. In addition to the ordinary cluster merge procedure, a specially designed merge procedure is developed to handle temporary clusters. A more detailed description is given in Section 5.6.

5.6. The Clustering Algorithm

5.6.1. Similarity Calculation

There are two kinds of similarity matrices maintained during the detection process. One is the story–story similarity matrix. Its purpose is to store the pairwise similarities between stories. The second one is the story-cluster similarity matrix. Its purpose is to store the similarity between stories and clusters.

The calculation of pairwise similarities of stories was discussed in Section 5.3. However, not all the stories in the story list are required to calculate the pairwise similarities. Only the stories in the first group in the story list are necessary to calculate pairwise similarities. For example, in Figure 3, we compute the similarities of story 1 with all the stories in the story list. We perform similar calculations up to story 24. The story−story similarity matrix will contain the similarity scores between the stories. The story-cluster similarity matrix will contain the similarity scores between the stories and the clusters.

During the merge process, some clusters are labeled as *real clusters*. At the end, the stories in the first group in the story list should be assigned to the real clusters. The event identification module, as described in Section 5.7, will determine the appropriate event for those stories.

5.6.2. Grouping Related Elements

After calculating the similarities between stories and clusters, the most similar elements are grouped together. The main steps of grouping related elements are described below:

(1) The story−story similarity matrix and the story-cluster similarity matrix store the similarity scores between stories and clusters. We first search for the highest score among the two matrices. This score indicates that, up to now, which two elements are the most similar. If the score comes from the story−story similarity matrix, it means that the corresponding two stories are very similar. If the score comes from the story-cluster similarity matrix, it means that the corresponding story and the real cluster are very similar.
(2) If the score is higher than a user-defined threshold, h, we invoke the combination algorithm. Otherwise, this module terminates.

The combination algorithm is described as follows:

(1) If the highest score comes from the story-cluster similarity matrix, we group the related story to the corresponding real cluster. Then, the content of the real cluster is updated by taking the average of the vectors in the cluster and the story.
(2) If the highest score comes from the story−story similarity matrix, we merge the corresponding two stories and we form a temporary cluster. To determine the representation of a new temporary cluster, we take the average of the respective vectors of the corresponding two stories as described in Section 5.2. The temporary cluster, T, will be further processed by either grouping to another temporary cluster or to an existing real cluster as follows:
    (a) We compare this new temporary cluster, T, to other existing temporary clusters and real clusters. The similarity measure was discussed in Section 5.3.
    (b) If the highest score exceeds the threshold, h, T is merged with the most similar element. The element can be a temporary cluster or a real cluster.
        - If another temporary cluster, T_a, is merged, a higher level temporary cluster, T_h, is formed. The content of T_h is found by computing the average of the corresponding vectors of T and T_a.

- If an existing real cluster, T_r is merged with T, we take the average of the corresponding vectors of T and T_r.

(c) If the highest score comes from the comparison between a story and a temporary cluster, the content of the temporary cluster is updated by taking the average of the vectors in the cluster and the story.

We include the temporary cluster in the story-cluster similarity matrix. Then, the main steps of grouping related elements are repeated. When it terminates, each story in the first batch of the story list becomes a seed of a new real cluster. The event identification module will be invoked.

5.7. Event Identification

The purpose of this module is to identify the associated events for the stories in the first batch of the story list. After the grouping related elements module, each story in the first batch should have been assigned to either a real cluster or a temporary cluster. Each temporary cluster contains a certain number of stories. Some of them belong to the first batch while some of them do not. We check each story in the temporary cluster and only stories which belong to the first batch are transferred to a real cluster. We compute the average of the corresponding vectors of the stories to be transferred in the temporary cluster. The resultant vectors are used to represent the content of the real clusters formed. The temporary clusters are then removed. At the end of this step, only real clusters remain and stories in the first batch will be associated with the appropriate real clusters.

The next step is to remove the stories in the first batch from the story list and receive stories from the next batch. Figure 4 depicts the structure of the story list after a detection cycle. The second batch of stories in the story list becomes the first batch of stories. If there exists a new batch, this batch is loaded and the stories in this batch become the last batch in the story list.

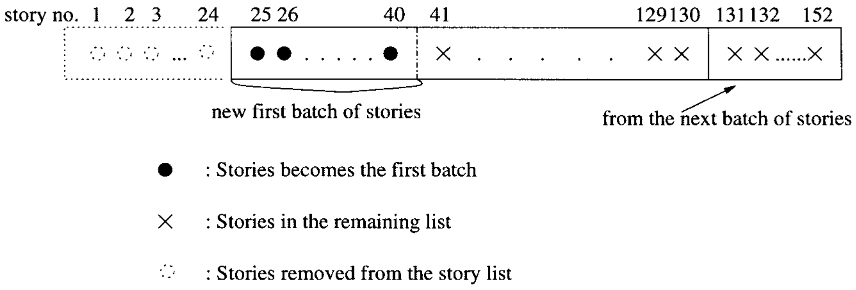

story no. 1 2 3 24 25 26 40 41 129 130 131 132 152
new first batch of stories
from the next batch of stories
● : Stories becomes the first batch
× : Stories in the remaining list
: Stories removed from the story list

Figure 4. Story list after a detection cycle.

6. EXPERIMENT, RESULTS, AND ANALYSIS

6.1. Evaluation Data and Methodology

We obtained the official development corpus of the TDT project from the Linguistic Data Consortium (LDC).[20] This corpus is annotated for 100 topics. The stories come from different sources as follows:

English sources:

- New York Times Newswire Service.
- Associated Press Worldstream Service (English content only).
- Cable News Network, "Headline News."
- American Broadcasting Company, "World News Tonight."
- Public Radio International, "The World."
- Voice of America, English news programs.

Chinese sources:

- Xinhua News Agency.
- Zaobao News Agency.
- Voice of America, Mandarin Chinese news programs.

The first four months of 1998 containing 43,666 stories were used in our experiments.

We follow the evaluation methodology specified in the TDT project.[21] In general, the performance of the detection system is based on the probabilities of misses and false alarms. A "miss" means that the target is not detected when it is actually present. A "false alarm" means that the target is falsely detected when it is actually not present. In other words, the event detection performance will be evaluated by measuring how well the stories belonging to each of the hypothesized event (event returned as the output of the system) matches with the stories belonging to the most likely evaluation event (event that has been manually annotated).

After our system processes all news stories, a set of hypothesized events are generated. We need to determine the link between each of the hypothesized events to an evaluation event. To determine the link, each evaluation event will be mapped to one hypothesized event that it best matches. The best match is defined as the match with the lowest detection cost. The detailed description of the match and the definition of the detection cost is given in Ref. 22. In the mapping task, the null event (i.e., the event with no stories) will be added to the set of hypothesized events to avoid nonsensical mappings. Note that different evaluation events may map to the same hypothesized event.

The calculation of miss and false alarm probabilities is done by the event-weighted method which computes the error probabilities for decisions so that all events have equal weights. Let E be the set of stories in the evaluation events, let E_b be the best match hypothesized event, and let S be the set of stories in the evaluation corpus being processed. The miss and false alarm probabilities computed by the event-weighted method are as

$$\begin{aligned} P_{\text{miss}} &= \frac{1}{N_{E_{\text{nonnull}}}} \sum_{E_{\text{nonnull}}} \{|E - E_b|/|E|\} \\ P_{\text{false alarm}} &= \frac{1}{N_E} \sum_{E} \{|E_b - E|/|S - E|\} \end{aligned} \tag{13}$$

where N_E is the total number of E, $|X|$ is the number of stories in the set X. These error probabilities are then combined into a single detection cost, C_{det} by assigning costs to miss and false alarm probabilities,

$$C_{\text{det}} = C_{\text{miss}} \cdot P_{\text{miss}} \cdot P_{\text{target}} + C_{\text{FA}} \cdot P_{\text{FA}} \cdot P_{\text{nontarget}} \tag{14}$$

where $C_{\text{miss}} = 1$ is the cost of a miss, $C_{\text{FA}} = 1$ is the cost of a false alarm, P_{miss} is the conditional probability of a miss, P_{FA} is the conditional probability of a false alarm, P_{target} is *a priori* target probability, $P_{\text{nontarget}} = 1 - P_{\text{target}}$.

Since C_{det} is associated with costs or errors, smaller values correspond to better performance. In TDT3, the official setting for P_{target} is 0.02. C_{det} is the bottom-line performance measure of the detection task. Nevertheless, the value of C_{det} is also a function of some application parameters. It is a function of the costs of detection errors and *a priori* target probabilities. Therefore, to provide a more meaningful measure of system performance, C_{det} will be normalized so that $(C_{\text{det}})_{\text{norm}}$ can be no less than one without extracting information from the source data. $(C_{\text{det}})_{\text{norm}}$ is defined as

$$(C_{\text{det}})_{\text{norm}} = \frac{C_{\text{det}}}{\min(C_{\text{miss}} \cdot P_{\text{target}}, C_{\text{FA}} \cdot P_{\text{nontarget}})} \tag{15}$$

As a result, the absolute value of $(C_{\text{det}})_{\text{norm}}$ is adopted for comparing performance in our experiment. Similar to C_{det}, the smaller the value, the better the performance.

6.2. Experiments on the Effects of Tuning the Parameter

A set of experiments were conducted to investigate the detection performance by varying the threshold h. We also varied the term weightings w_s and w_c for calculating the similarity score. We chose the top 50 story terms, the top 15 named entities, and the top 15 concept terms to represent a story.

Figure 5 shows the detection performance measured in $(C_{\text{det}})_{\text{norm}}$ when we varied the threshold, h. The results show that the best $(C_{\text{det}})_{\text{norm}}$ of 0.3248 is obtained when the threshold h is 0.14. In general, the system can achieve relatively small false alarm probability. One reason is that we always compare the stories in the first batch to existing clusters so that similar stories can be grouped as far as possible. However, the undesirable effect is that relatively large miss probabilities are obtained. Figure 6 shows the detection performance

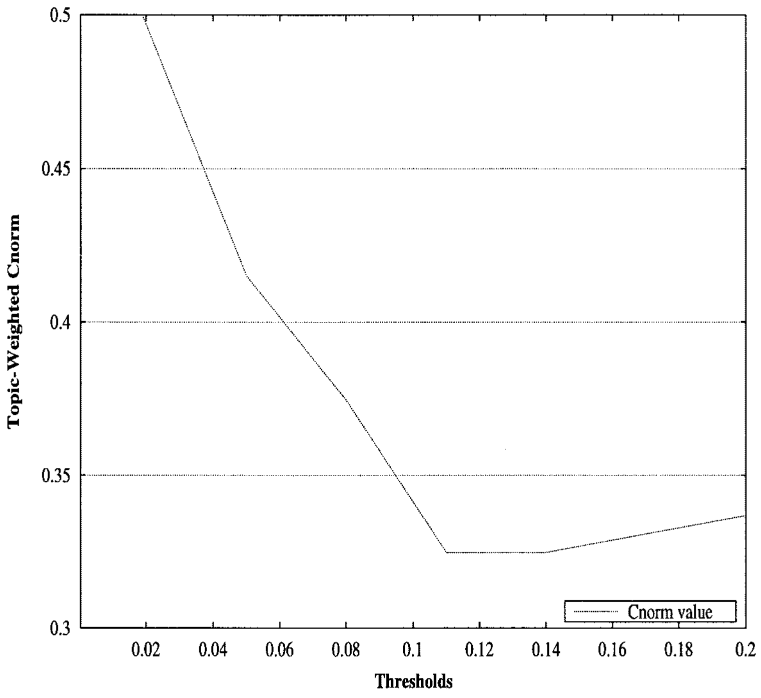


Figure 5. The detection performance of different threshold, h.

when we varied w_c and we fixed w_s at 0.5. The result shows that the best values of w_s and w_c are both equal to 0.5.

6.3. Experiments on the Effects of Concept Terms

The purpose of this experiment is to investigate the effect of concept terms on the detection performance. We used the same development set as previous experiments. We chose the top 50 story terms, the top 15 named entities, and the top 15 concept terms (if they are used) to represent a story. The parameters w_s, w_c are all set to 0.5. The threshold h is varied to test a range of values. We ran the experiments on two conditions. The first condition corresponds to a representation where all concept terms, named entities, and story terms are used. The second condition is similar to the first condition except for the fact that concept terms are not used.

Figure 7 shows the detection performance for different conditions. The result shows that using concept terms can improve the detection performance. The combination of features provides more useful characterization of the stories and thus a better result is obtained.

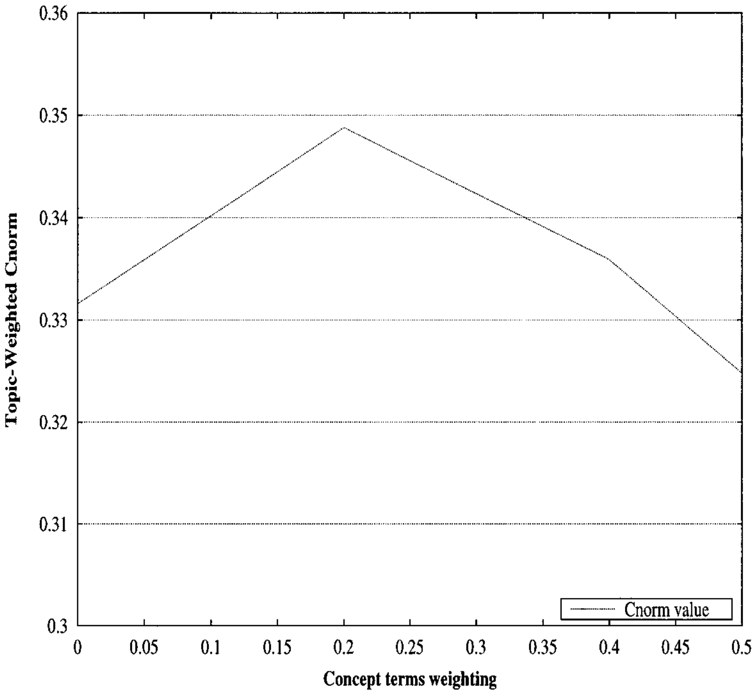


Figure 6. The detection performance of different concept term weightings w_c.

7. CONCLUSIONS AND FUTURE WORK

The goal of event detection is to develop an intelligent system to detect events automatically from an incoming source of news stories. Traditionally, each story or event is represented by a set of story keywords and the corresponding term weights. In our proposed system, each story or event is represented by a combination of three kinds of features, namely, concept terms, named entities, and story terms. Concept terms are extracted from a concept database according to related stories in a concept generation corpus against a sentence. Named entities are extracted from the story by a transformation-based error-driven part-of-speech tagger.

Gross translation on Chinese stories is performed so that each Chinese story is represented by a set of English words. Then, we perform agglomerative clustering to group related stories together. Each cluster formed is represented as an event. The aim of our clustering method is to group related stories in the window of deferral period as much as possible. Based on the characteristics of temporary cluster and real cluster, similar stories and temporary clusters are merged to form a higher level temporary cluster.

Experimental result shows that our clustering approach can minimize the false alarm probability. The result also demonstrates the strength of the combined feature representation over the traditional story keyword representation.

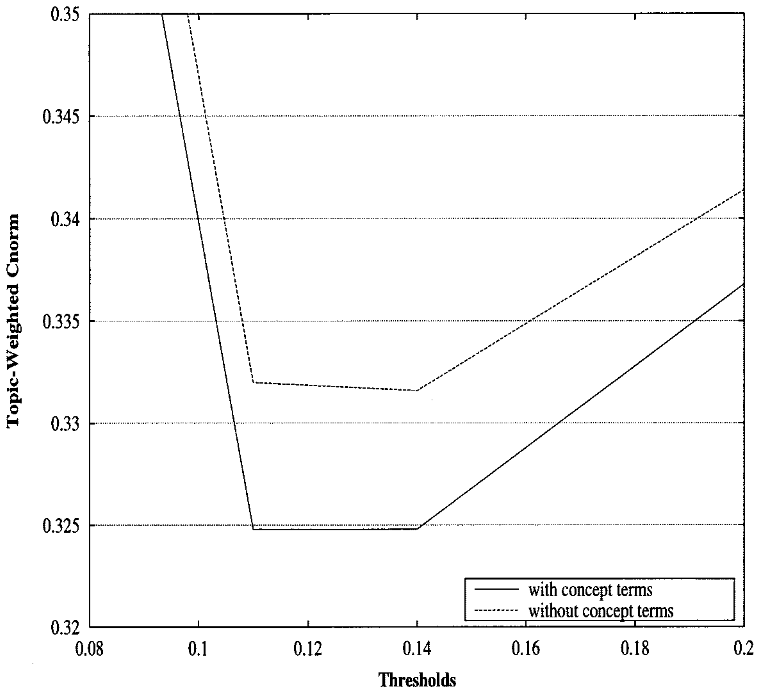


Figure 7. The detection performance of different combinations of terms.

The concept terms can provide useful characterization of the stories and they bring benefits for the detection task.

Although the system performs quite well, it has some limitations. Our next goal is to further study the limitations and to improve the system performance. One future work is to develop a more powerful translation module. The translation algorithm can be improved so that each Chinese word can be translated into a set of more meaningful English terms. Another area for investigation is to consider incremental inverse document frequency which has been shown to be useful in other information retrieval systems. Finally, a more sophisticated clustering algorithm can be developed to reduce the miss probability.

References

1. Allan J, Carbonell J, Doddington G, Yamron J, Yang Y. Topic detection and tracking pilot study final report. In: Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop, 1998.
2. Allan J, Lavrenko V, Malin D. Detections, bounds and timelines: Umass and TDT-3. In: The DARPA Topic Detection and Tracking Workshop, February 2000, p 167–174.
3. Allan J, Papka R, Lavrenko V. On-line new event detection and tracking. In: Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Association for Computing Machinery Special Interest Group of Information Retrieval, 1998, p 37–45.

4. Chen H, Houston AL, Sewell RR, Schatz BR. Internet browsing and searching: User evaluation of category map and concept space techniques. J Amer Soc Inf Sci 1998; 49(7):582–603.
5. Chen HH, Ku LW. Description of a topic detection algorithm. In: The DARPA Topic Detection and Tracking Workshop, February 2000, p 165–166.
6. Chen H, Martinez J, Kirchhoff A, Ng TD, Schatz BR. Alleviating search uncertainty through concept associations: Automatic indexing, co-occurrence analysis and parallel computing. J Amer Soc Inf Sci 1998;49(3):206–216.
7. Chen H, Martinez J, Ng TD, Schatz BR. A concept space approach to addressing the vocabulary problem in scientific information retrieval: An experiment on the worm community system. J Amer Soc Inf Sci 1997;49(1):17–31.
8. Franz M, McCarley JS, Ward T, Zhu WJ. Segmentation and detection at IBM: Hybrid statistical models and two-tired clustering. In: The DARPA Topic Detection and Tracking Workshop, February 2000, p 149–154.
9. Houston AL, Chen H, Schatz BR, Sewell RR, Tolle KM, Doszkocs TE, Hubbard SM, Ng TD. Exploring the use of concept space, category map techniques and natural language parsers to improve medical information retrieval. In: Decision Support Systems, Special Issue on Decision Support for Health Care in a New Information Age, January, 1997.
10. Leek T, Jin H, Sista S, Schwartz R. The BBN crosslingual topic detection and tracking. In: The DARPA Topic Detection and Tracking Workshop, February 2000, p 123–128.
11. Meng H, Ip CW. An analytical study of transformational tagging for Chinese text. In: Research on Computational Linguistics Conference (ROCLING XII), Taipei, Taiwan, Republic of China, 1999, p 101–122.
12. Ponte JM, Croft WB. Text segmentation by topic. In: Proceedings of the First European Conference on Research and Advanced Technology of Digital Libraries, 1997, p 120–129.
13. Porter MF. An algorithm for suffix stripping. Program 1980;14(3):130–137.
14. Rasmussen E. Clustering algorithms. In: Frakes WB, Baeza-Yates R, editors. Information retrieval. Englewood Cliffs, NJ: Prentice Hall; 1992.
15. Wong KL, Lam W, Yen J. Interactive Chinese news event detection and tracking. In: The Second Asian Digital Library Conference, 1999, p 30–43.
16. Xu J, Croft WB. Query expansion using local and global document analysis. In: Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1996, p 4–11.
17. Yamron JP, Gillick L, Knecht S, Lowe S, van Mulbregt P. Statistical models for tracking and detection. In: The DARPA Topic Detection and Tracking Workshop, February 2000, p 139–144.
18. Yang Y, Pierce T, Ault T, Carbonell J. Combining multiple learning strategies to improve tracking and detection performance. In: The DARPA Topic Detection and Tracking Workshop, February 2000, p 129–134.
19. Yang Y, Pierfce T, Carbonell J. A study on retrospective and on-line event detection. In: Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1998, p 28–36.
20. http://morph.ldc.upenn.edu/TDT Linguistic Data Consortium.
21. http://morph.ldc.upeen.edu/TDT/Guide/manual.front.html 1999 Topic Detection and Tracking TDT3 Task Definition and Evaluation Plan.
22. http://morph.ldc.upenn.edu/TDT Topic Detection and Tracking Phrase 2 (TDT2) Evaluation Plan, 1998.