Hot Topic Extraction Based on Timeline Analysis and Multidimensional Sentence Modeling

Kuan-Yu Chen, Luesak Luesukprasert, and Seng-cho T. Chou

Abstract—With the vast amount of digitized textual materials now available on the Internet, it is almost impossible for people to absorb all pertinent information in a timely manner. To alleviate the problem, we present a novel approach for extracting hot topics from disparate sets of textual documents published in a given time period. Our technique consists of two steps. First, hot terms are extracted by mapping their distribution over time. Second, based on the extracted hot terms, key sentences are identified and then grouped into clusters that represent hot topics by using multidimensional sentence vectors. The results of our empirical tests show that this approach is more effective in identifying hot topics than existing methods.

Index Terms—Aging theory, clustering, hot topic detection, term weighting, topic detection and tracking.

1 Introduction

Although timely access to information is becoming increasingly important in today's knowledge-based economy, gaining such access is no longer a problem because of the widespread availability of broadband in both homes and businesses. Ironically, high-speed connectivity and the explosion in the volume of digitized textual content available online has given rise to a new problem, namely, information overload. Clearly, the capacity for humans to assimilate such vast amounts of information is limited. Topic Detection has emerged as a promising research area that harnesses the power of modern computing to address this new problem. Topic Detection is a subprocess of Topic Detection and Tracking (TDT) that attempts to identify "topics" by exploring and organizing the content of textual materials, thereby enabling us to aggregate disparate pieces of information into manageable clusters automatically. In the context of news, Topic Detection can be viewed as an event detection that groups stories into a corpus, wherein each group represents a single topic.

In this paper, we are only interested in extracting "hot topics" from a given set of text-based news documents published during a given time period. Our aim is to alleviate the information overload problem by focusing on important topics, that is, topics that appear with abnormally high frequency during a specified time period and typically contain several "hot terms" that are the basis of topic extraction. We introduce two critical properties of a hot term, "pervasiveness" and "topicality," which can improve the quality of the results from the extraction process.

• *The authors are with the Department of Information Management, National Taiwan University, 85 Section 4 Roosevelt Road, Taipei, 106 Taiwan (ROC).*
E-mail: {r92725001, chou}@ntu.edu.tw, kim@myfamilylink.net.

Manuscript received 1 Feb. 2006; revised 6 Nov. 2006; accepted 13 Feb. 2007; published online 21 Mar. 2007.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0050-0206.
Digital Object Identifier no. 10.1109/TKDE.2007.1040.

The remainder of this paper is organized as follows: In Section 2, we define key concepts and terms and introduce works that are directly related to the ideas presented in this paper. Section 3 describes our novel approach for recognizing hot terms and clustering hot topics via multidimensional sentence modeling. In Section 4, we demonstrate the superiority of our approach with comparative empirical results. Finally, in Section 5, we present our conclusions and some future research directions.

2 Previous Works

2.1 What Is a Topic?

A topic is defined as a seminal event or activity, along with all directly related events and activities [23]. Thus, we can infer that a topic consists of events and activities, both of which are defined in greater detail in [22]. A TDT event is defined as something that happens at a specific time and place, along with all the necessary preconditions and unavoidable consequences [22]. Such an event might be a car accident, a meeting, or a court hearing. A TDT activity is a connected series of events with a common focus or purpose that happens in specific places during a given time period [22]. For instance, a TDT activity may be an election campaign, an investigation, or a disaster relief effort.

2.2 What Is a Hot Topic?

In [13], a "hot topic" is defined as a topic that appears frequently over a period of time. The "hotness" of a topic depends on two factors: how often hot terms appear in a document and the number of documents that contain those terms. Moreover, no topic can remain hot indefinitely; in other words, every topic goes through a life cycle of birth, growth, maturity, and death. Hence, the "hotness" of each topic evolves over a given period of time. In the case of news, topics have different levels of popularity or "hotness." Some are so hot that every news channel broadcasts them and reports on them in great detail, whereas others that are not so popular are only reported by a few channels. Regardless of the peak level of "hotness," news topics

eventually "cool off" and are replaced by other more up-to-date stories.

In summary, in the context of news, a "hot" topic T has the following characteristics:

- It appears in many news stories on a news channel.
- It appears on many news channels.
- It has strong continuity, which means that many different events related to T are also reported.
- It varies in popularity over time.

2.3 Topic Detection

A term weighting scheme can be used to capture important or representative terms that feature in the content of a document. There are many ways to evaluate the significance of a term, ranging from simply identifying its existence to evaluating its distribution level in a document or in a whole corpus [6], [7], [12], [20], [21]. To identify topics in large sets of documents, we have to determine the key terms that sufficiently describe the topics.

The most common term weighting scheme for processing index terms is TF-IDF [5], which stands for term frequency —inverse document frequency. In this scheme, if a word w occurs in several documents in a corpus, it may be a common term that can be used to identify topics. Since the TF-IDF scheme emphasizes the importance or uniqueness of each term, it only identifies terms that occur in a few documents in a corpus. For hot topic extraction, however, terms that appear in a large number of documents in a corpus must be identified. To do this, Bun and Ishizuka [13] proposed a different term weighting scheme TF-Proportional Document Frequency (TF*PDF), which assigns greater weights to terms that occur frequently in many documents on many channels and lower weights to those that are rarely mentioned. Although TF*PDF captures the basic concept of a hot topic, its weakness is that it does not consider variations in the popularity of a topic over time. Even so, the scheme offers an excellent foundation on which to build our model.

2.4 Aging Theory

Capturing variations in the distribution of key terms on a time line is a critical step in extracting hot topics. Therefore, it is essential to track the terms to determine what stage of their life cycle they are in. Previous research works have recognized that topics in a continuous stream of documents can be identified by a simultaneous temporal burst of related documents [9], [10]. Chen et al. [2] applied the Aging Theory to model a news event's life span and suggested that a news event can be considered as a life form that goes through a life cycle of birth, growth, decay, and death, reflecting its popularity over time. They utilized the concept of energy to track the life cycles of events. The level of energy indicates the stage of a news event in its life span. The energy of an event increases when the event becomes popular and decreases as its popularity wanes. Hence, the Aging Theory is suitable for tracking the variations in the frequency of terms, which we consider critical to a successful hot topic extraction.

2.5 Sentence Modeling

Using a traditional vector space model to compare the similarity of two documents is a common practice in the field of TDT. Specifically, in New Event Detection and Topic Tracking, the cosine similarity measure is used to judge whether an incoming document relates to a new event or is similar to an existing event. Although there are many language tracking and modeling methods based on machine learning, thus far, the vector space model has achieved the best results [6]. However, its limitations are evident when we set a higher precision or recall rate for TDT work. Based on the traditional similarity model, a sentence vector contains too few words to provide sufficient hints for measuring similarity because a sentence has far fewer key terms to represent the central idea than a paragraph or an entire document [16]. Previous research has demonstrated that it is hard to compare the similarity of sentences when there are insufficient keywords in the sentences being compared, which is a key defeating factor in news event detection [8].

The shortcomings of the simple vector space model approach suggest that there is a need for document representation combined with existing cosine similarity metrics [6]. Our sentence model is a refined representation of the traditional simple vector model that describes the detailed characteristics of a sentence [13]. This new model utilizes some key characteristics of an event, such as who was involved in the event and where it occurred. Thus, named entities (NEs) or noun phrases play an important role in identifying the meaning of a sentence or document. There are several state-of-the-art approaches focusing on the special treatment of NEs or noun phrases [6], [8], [17], [18], [12], [25]. In addition to NEs and noun phrases, the relationship between each term in a sentence vector can also be utilized to enrich the vector and thereby improve the clustering of textual material [1].

3 Model Development

As noted in Section 2, the inadequacy of current techniques for hot topic extraction is due to the fact that 1) existing approaches for extracting key terms do not consider a term's life cycle and 2) traditional sentence modeling techniques are limited in their ability to cluster useful information. We now present solutions to these problems.

3.1 Hot Term Recognition with Timeline Analysis

We have already observed that simply considering the frequency of hot terms is insufficient for hot topic detection. Since terms or words are the basic elements of any news report, changes in the content of reports will be reflected by variations in a term's usage. Because a topic is composed of many related events, changes in a topic's popularity are therefore accompanied by variant usage of key terms or "hot terms." Our premise is therefore that it is necessary to consider variations in the frequency that terms are used over time in order to accurately identify hot topics.

Hot terms have the following properties:

- *Pervasiveness*. This property refers to the frequency with which a term appears in a set of documents. The higher the frequency is, the more pervasive a term becomes.
- *Topicality*. This property refers to the variation in the frequency of usage of a term over time. A term is more topical if its usage varies greatly.

We define the weight of a hot term as the sum of the weights given by each of these two properties. The process

of determining hot terms and their weights consists of three steps.

3.1.1 Determine the Traceable List

Since it is computationally expensive to track all the terms that appear in all news stories, we focus on a subset of terms called a "traceable list." First, a stop list is used to remove stop words from the set of terms, which is followed by a stemming process based on Porter's stemming algorithm [14]. We then rank each term based on its frequency and select terms with frequencies above a specified threshold.

3.1.2 Determine the Topical Property of Each Term in the Traceable List

This step is to track the life cycle of the terms in the traceable list. Specifically, it is to calculate the variation in the "life support" value of each term on the list. We make use of four functions [2] for this purpose, namely, getEnergy(), energyFunction(), energyDecay() and getVariation(). Function getEnergy() calculates the nutrition (energy) that a term receives at a specific time slot. energyFunction() converts energy into a life support value, which decays over time, as captured by energyDecay(). The variance of a term's life support value is computed using the getVariation() function, which quantifies how topical a term is. The procedure below sums up the steps involved, and function details follow:

$$\begin{aligned}&\text{For each timeslot } s \text{ of term } t\\ &\quad \text{oldEnergy} = \text{energyFunction}^{-1}(\text{lifeSupport}_{t,s-1})\\ &\quad \text{newEnergy} = \text{t.getEnergy}(t,s)\\ &\quad \text{lifeSupport}_{t,s} = \text{energyFunction}(\text{oldEnergy} + \text{newEnergy})\\ &\quad \text{lifeSupport}'_{t,s} = \text{energyDecay}(\text{lifeSupport}_{t,s})\\ &\text{Loop}\\ &\text{t.getVariation}()\end{aligned} \tag{1}$$

1. *getEnergy().* The energy of a term t is defined as the degree of association between t at time slot s over the channels. As revealed by the formula, term t is given a higher value if it appears more frequently on a specific channel in the specified time slot s than in any other time slots. The total energy level of t in s is the sum of the values from all channels. Hence, hot terms are those that have high energy in all channels.

   Input: a single unit of time slot s, the given term t

   Output: the energy of t in s

$$\mathrm{E}_{ts} = \mathrm{nf} * \left(\sum_{c \in \mathrm{C}} \chi^2_{t,c}\right), \tag{2}$$

   where

   - C is the set of channels,
   - $\chi^2_{t,c}$ is the association between term t and the time slot s in channel c, given by the following equation from [18]:

$$\frac{(\mathrm{A}+\mathrm{B}+\mathrm{C}+\mathrm{D}) * (\mathrm{AD}-\mathrm{BC})^2}{(\mathrm{A}+\mathrm{B}) * (\mathrm{C}+\mathrm{D}) * (\mathrm{A}+\mathrm{C}) * (\mathrm{B}+\mathrm{D})}$$

	ln Channel i	ln other channel
$t \in s$	A	B
$t \notin s$	C	D

   where

   - A is the number of occurrences of t on channel i during time slot s,
   - B is the number of occurrences of t on other channels in time slot s,
   - C is the number of occurrences of t on channel i when t is not in time slot s, and
   - D is the number of occurrences of t on other channels when t is not in time slot s.

   - E_{ts} is the energy value of term t in s, and
   - nf is the nutrition transfer factor [2], which is an empirical constant.

   Note that the nutrition transfer factor is used to increment the energy of a term based on findings from a new input. The nutrition decay factor [2] below works in the opposite direction to provide the mechanism by which the energy of a term can be periodically reduced over time.

2. *energyFunction().* We define our energy function as a natural logarithm of the accumulated energy value of the term as a way of converting a term's energy into its life support value. Its inverse function returns the previously accumulated energy value of the term in the previous time slot.

   Input: the energy value of the term t in the given time slot s

   Output: the life support value of t in s

$$\text{life support}_{t,s} = \ln \mathrm{eng}_{t,s}, \tag{3}$$

   where

   - $\mathrm{lifesupport}_{t,s}$ is the life support value of t in time slot s and
   - $\mathrm{eng}_{t,s}$ is the accumulated energy value of t so far.

3. *energyDecay().* Over time, a term's life support value gets modified by a decay factor that represents the decay in each time slot. If the decayed life support value is negative, then it is set to 0.

   Input: the life support value of the term t in the given time slot s

   Output: the decayed life support value

$$\begin{aligned}&\text{In each time slot}\\ &\text{lifesupport}'_{t,s} = \text{lifesupport}_{t,s} - \mathrm{d}\\ &\text{If lifesupport}'_{t,s} < 0\\ &\quad \text{lifesupport}'_{t,s} = 0,\end{aligned} \tag{4}$$

   where

   - d is the decay nutrition factor [2], which is an empirical constant,

- $\text{lifesupport}_{t,s}$ is the life support value before decay, and
- $\text{lifesupport}'_{t,s}$ is the life support value after decay.

4. *getVariation().* The get variation function calculates the variance of the life support values of term t by using a standard variation formula.

   Input: the term t, an interval I

   Output: the variation in the life support value of the term t

$$\text{V}_t = \sqrt{\frac{1}{\text{N}}\sum\left(\text{lifesupport}_{t,s} - \overline{\text{lifesupport}}\right)^2}, \quad (5)$$

   where

   - N is the number of time slots in the given interval I,
   - $\text{lifesupport}_{t,s}$ is the life support value in each time slot,
   - $\overline{\text{lifesupport}}$ is the average life support value, and
   - V_t is the variation in the life support values of t during I.

3.1.3 Assign a New Weight to Each Term

This procedure involves several steps.

For each term t:

1. Get the original TF^*PDF weight of t which is calculated by the following equation from [13]:

$$W_j = \sum_{c=1}^{c=|C|} |F_{jc}| \exp\left(\frac{n_{jc}}{N_c}\right)$$
$$|F_{jc}| = \frac{F_{jc}}{\sqrt{\sum_{k=1}^{k=K} F_{kc}^2}}, \quad (6)$$

   where

   - W_j is the weight of term j,
   - W_j is the weight of term j,
   - F_{jc} is the frequency of term j in channel c,
   - n_{jc} is the number of documents in channel c where term j occurs,
   - N_c is the total number of documents in channel c,
   - k is the total number of terms in channel c, and
   - |C| is the number of channels.

   The TF^*PDF weighting scheme consists of three components:

   - *Summation.* The final weight is the summation of term weights gained from each news channel.
   - F_{jc}. F_{jc} represents the term frequency. Because of the different archive size of each news channel, it is reasonable to normalize F_{jc} to give equal importance to the same term from each channel.
   - $\exp\left(\frac{n_{jc}}{N_c}\right)$. This component is the implementation of PDF. It is the exponential growth of the number of documents containing the term compared to the total number of documents in the channel.

2. Get the rank or frequency order (FO) of t by sorting the terms in descending order based on their weights. The term that appears most frequently (with the highest TF^*PDF weight) receives a $\text{FO} = 1$, whereas the term with the least frequency receives the largest FO value.
3. Get the rank or variance order (VO) of t by sorting the terms in descending order based on their variance in life support values. The term with the largest variance receives a $\text{VO} = 1$, whereas the term with the smallest variance receives the largest VO value.
4. Calculate the bonus point (BP) by taking the difference between FO and VO. The BP is used to increment or decrement the final weight of a term. BP is positive when VO is small, meaning that the usage of t varies greatly, suggesting that it may be a topical term:

$$\text{BP} = \text{FO} - \text{VO}. \quad (7)$$

5. Calculate the weight modifier rate of t:

$$\text{Weight Modifier} = \frac{\text{BP}}{\text{Total Number of Terms in the Traceable List}}. \quad (8)$$

6. Calculate the new weight of t:

$$\text{New Weight} = \text{TFPDF}_t + \text{Var}_t * (1 + \text{Weight Modifier}), \quad (9)$$

   where

   - TFPDF_t is the TF^*PDF Weight of t and
   - Var_t is the variation in life support values of t.

   The new weighting scheme has two parts:

   1. TFPDF_t, which checks if a term is pervasive, and
   2. Var_t^* $(1 + \text{Weight Modifier})$, which gives bias to more topical terms.

Each term is assigned a new term weight by the above term weighting scheme. Then, the traceable list is ordered and the top-ranked k terms are chosen to form the Hot Term Recognition List. By selecting the top-ranked k terms, we are not making any assumption about the number of topics in the corpus. k is merely an adjustable threshold that helps to limit the number of topics returned as a result of the extraction process. If k is small, then only the hottest topics will be extracted, and vice versa.

3.2 Multidimensional Sentence Modeling

With the hot terms extracted and their weights assigned, we are ready to generate "hot sentences." Let the weight of a sentence be the average of the weights of all the terms that it possesses. We rank the sentences by their weights, and the top-ranked k sentences will be considered hot and therefore used in the subsequent sentence clustering process. To avoid biased domination, those sentences with too few terms (guarded by an adjustable threshold) or without any hot terms will not be considered as hot sentences. Hot terms

TABLE 1
Hot Terms Extracted by $\mathrm{TF}^{*}\mathrm{PDF}$ (Pervasive Only)

Order										
1~5	u.s.	0.756934	who	0.522119	will	0.499683	state	0.459315	govern	0.457419
6~10	offici	0.438817	peopl	0.398556	year	0.3978	kill	0.369112	presid	0.35723
11~15	countri	0.341659	new	0.336509	two	0.331194	on	0.326028	unit	0.32055
16~20	sai	0.30922	last	0.302004	polic	0.299726	forc	0.296029	i	0.290644
21~25	report	0.286867	told	0.267419	iraq	0.263157	minist	0.263051	attack	0.260654
26~30	leader	0.252087	secure	0.248534	polit	0.244996	troop	0.239793	nation	0.236034

are those in the Hot Term Recognition List mentioned above.

Next, we apply the hierarchical agglomerative clustering (HAC) [4] technique to group the hot sentences into clusters. However, instead of adopting the traditional one-vector model used by previous studies, we propose a multidimensional sentence vector model that considers more diverse criteria for identifying sentence similarity. The five vectors used are listed as follows:

1. *Hot Term Vector (HTV).* An HTV is composed of hot terms generated by the aforementioned hot term generation process. This is the basic vector used by previous researchers [4].
2. *Named Entity Vector (NEV).* A NEV is made up of NEs contained in a sentence. We mainly considered the event, language, location, nationality, organization, and person NEs in this study [3].
3. *Concept vectors (CVs).* Direct CV (DCV), Kind of Vector (KV), and Part of Vector (PV) are CVs derived from the background knowledge of each hot term w contained in HTV, and we selected WordNet [24] as the source of background knowledge on words:
   - DCV: Each element contains the synonyms of w obtained from WordNet.
   - KV: Each element contains the hypernyms of w taken from WordNet. Since there can be more than one level of hypernyms for w, we choose to take the first level only.
   - PV: Each element contains the regular homonyms of w listed on WordNet. As there may be more than one homonym for w, we choose the first concept in the ordered list returned by WordNet, which orders homonyms by how often they are used in "standard" English [1].

Then, the similarity of any two sentences S1 and S2 is defined as the weighted sum of the cosine similarities of corresponding vectors:

$$\begin{aligned}\mathrm{Similarity}(\mathrm{S1},\mathrm{S2}) &= \mathrm{a} * \cos(\mathrm{HTV}_1, \mathrm{HTV}_2) \\ &+ \mathrm{b} * \cos(\mathrm{NEV}_1, \mathrm{NEV}_2) + \mathrm{c} * \cos(\mathrm{DCV}_1, \mathrm{DCV}_2) \\ &+ \mathrm{d} * \cos(\mathrm{PV}_1, \mathrm{PV}_2) + \mathrm{e} * \cos(\mathrm{KV}_1, \mathrm{KV}_2),\end{aligned} \tag{10}$$

where

- a, b, c, d, and e are the weights assigned to the cosine similarities, where $\mathrm{a} + \mathrm{b} + \mathrm{c} + \mathrm{d} + \mathrm{e} = 1$,
- similarity (S1, S2) is the total similarity value of Sentence 1 and Sentence 2, and
- cos (Vector 1, Vector 2) is the cosine similarity of Vector 1 and Vector 2.

4 Empirical Verification

We implemented a Java-based Hot Topic Extraction System that can be easily deployed on any Java virtual machine (JVM) platform and gathered news reports from several online news channels as our data source. The experiments tested the viability of the two major components in our work, namely, our new term weighting scheme and our new sentence-modeling scheme.

4.1 Data Source

Our data was gathered from the world news categories of several online news channels between 5 March 2005 and 10 April 2005. We collected 664 news reports from the Washington Post, 794 reports from Reuters, and 578 reports from CNN. Each news report on a particular event was saved as a text file. Our data collection period was limited to just one month because we were not trying to identify long-term patterns in the data. Instead, our focus was to explore the effectiveness of our approach to identify hot topics, which are, by nature, short lived. Hence, there was no need to collect vast amounts of data spread over a long time horizon.

4.2 Term Weighting Analysis

For this analysis, we conducted two experiments. First, we compared hot terms extracted by the $\mathrm{TF}^{*}\mathrm{PDF}$ weighting scheme with our proposed method. This experiment validates the effectiveness of our method over $\mathrm{TF}^{*}\mathrm{PDF}$. In the second experiment, we demonstrated how we can identify genuine hot terms by tracking each term's life cycle.

4.2.1 Experiment 1 (Comparison with $\mathrm{TF}^{*}\mathrm{PDF}$)

Tables 1 and 2 show the extracted hot terms by using $\mathrm{TF}^{*}\mathrm{PDF}$ and our new scheme, respectively. The hot terms extracted by tracking their life cycles (Table 2) differ from those extracted by the $\mathrm{TF}^{*}\mathrm{PDF}$ weighting scheme (Table 1), especially on NEs. Table 2 contains more NEs than Table 1. In fact, most of the 30 top-ranked terms in Table 2 are NEs or noun phrases.

TABLE 2
Hot Terms Extracted by Tracking Their Life Cycles (Pervasive + Topical)

Order										
1~5	saudi	2.580876	maskhadov	2.548929	nichol	2.349703	taliban	2.326782	haradinaj	2.286649
6~10	tung	2.250932	berlusconi	2.157237	kosovo	2.073946	ira	2.062263	kong	2.053062
11~15	hong	2.037942	Sharon	1.973848	eu	1.893799	chen	1.889052	energi	1.886383
16~20	sgrena	1.884283	feder	1.814742	mesa	1.81418	delai	1.810086	korean	1.807991
21~25	brother	1.794279	cabinet	1.730998	checkpoint	1.648392	abba	1.646569	shiit	1.636939
26~30	north	1.626789	u.s.	1.618159	russian	1.614304	africa	1.582529	taiwan	1.551424

As noted earlier, a hot term must be "pervasive" and "topical." However, TF*PDF only considers the "pervasive" property. As shown in Fig. 1, there is a strong positive correlation between TF and PDF. Thus, the TF*PDF weighting scheme focuses on extracting terms of high frequency and not on terms that really describe a story, an event, or a topic. Although we have applied a stop list to filter out many of the auxiliary terms in Table 1, it appears that some of them have returned as a result of stemming. In contrast, Table 2, which is based on our algorithm, shows that most of the extracted terms are NEs or noun phrases. Although some noise exists in the hot terms extracted by our algorithm, the result is much more appealing than that derived by the TF*PDF algorithm. It is also worth noticing that our algorithm removed most of the auxiliary terms, even after stemming.

4.2.2 Experiment 2 (Lifecycle-Based Identification of Hot Terms)

In Fig. 2, we compare the NE "maskhadov" in Table 2 with the following terms:

- "maskhadov." The name's energy value jumped significantly on the first day and hit a plateau before dropping to the bottom on days 6 and 7. Afterwards, it hit a peak of 3.833 but suddenly dropped again.
- "want," "day" (the prestemmed form of "dai"), "back," and "until." These terms had lower variations in their life support values, fluctuating in the range of 0 to 2.

Because NEs go with events, we can see that the term "maskhadov" varies dramatically over the selected period. Its life support value on days 2, 5, and 9 was higher because of its strong association with events that occurred on those days. In fact, there were many news reports about Maskhadov, a Chechen hero killed by Russian forces, between days 2 and 9. Therefore, we know that the person "maskhadov" featured prominently in news reports over that period by tracking the name's life cycle.

In contrast, we cannot determine what happened in the same period from the terms "want," "day," "back," and "until," which do not provide a reference to any particular context by themselves.

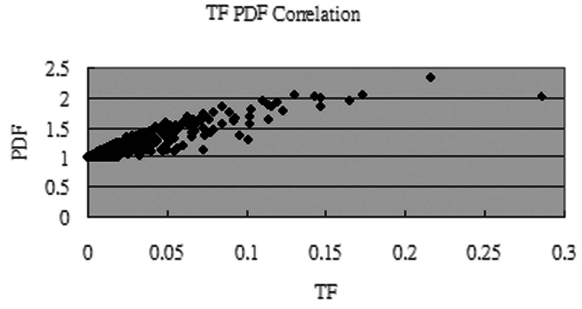


Fig. 1. The correlation between TF and PDF.

4.3 Sentence Clustering Analysis

We use two criteria to evaluate the performance of our clustering algorithm:

- *Coverage rate (CR).* This indicates how many hot topics are extracted for a specified period:

$$\text{Coverage Rate (CR)} = \frac{\text{Extracted Hot Topics}}{\text{Actual Hot Topics}} \times 100\%. \tag{11}$$

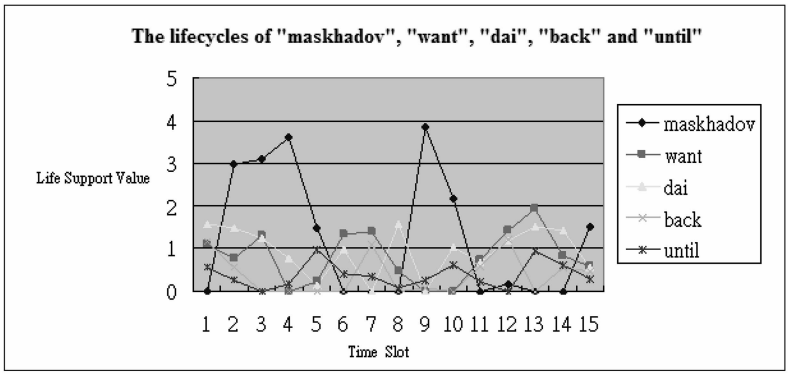
- *Correctness.* This checks the quality of our final clusters by using the following criteria, borrowed from [13]:
  1. *Miss Sentence (MS).* The sentence has been clustered to a topic but incorrectly.
  2. *Fail Sentence (FS).* The sentence belongs to a cluster but has not been clustered in.
  3. *Not Clustered Sentence (NC).* The sentence does not belong to any topic cluster.

Again, we conducted two experiments to evaluate the performance of our approach.

4.3.1 Experiment 1 (Coverage Rate)

To assess the CR, we first manually categorized each news report into topics and calculated the percentage of documents in each topic. The goal of this experiment was to compare the hot topics extracted against genuine hot topics in the source corpus. Table 3 shows the list of topics compiled by manual categorization. Note that more than 56 percent of the topics discovered in the corpus had less than five news stories; hence, we excluded them from our hot topics list.

With the baseline established, we could test the results of our hot topic extraction process. We first analyzed the CR.



Term	3/7	3/8	3/9	3/10	3/11	3/12	3/13	3/14	3/15	3/16	3/17	3/18	3/19	3/21	V
maskhadov	0	2.98	3.10	3.59	1.48	0	0	0	3.83	2.17	0	0.17	0	0	1.50
back	1.15	0.55	0	0	0	0	1.09	0	0.10	0	0.58	1.22	0	0.57	0.46
until	0.55	0.27	0	0.17	0.96	0.41	0.35	0.08	0.28	0.62	0.23	0	0.96	0.62	0.30
dai	1.56	1.47	1.24	0.77	0.14	0.97	0	1.58	0	1.05	0.69	1.19	1.50	1.42	0.55

Fig. 2. The life cycles of "maskhadov," "want," "dai," "back," and "until."

TABLE 3
Baseline Identification of Hot Topics and the Percentage of News Stories for Each Topic

News Topics	# of Related Stories	Percentage
Rice	25	3.10%
The relationship between China and Taiwan	23	2.85%
Syrian troops quit north Lebanon	21	2.61%
Conflicts in West Bank	21	2.61%
Bolivian president resigns amid protests	16	1.99%
HK Leader to Announce Resignation	16	1.99%
Italian Intelligence Agent Killed in Baghdad	14	1.74%
Kosovo PM Indicted for War Crimes	14	1.74%
The nuclear weapons in North Korea	14	1.74%
Israel and Palestinian	12	1.49%
The nuclear weapons in Iran	12	1.49%
Madrid train bomb	11	1.36%
Massive turnout at pro-Syria rally	11	1.36%
U.N. troops quit Iraq	11	1.36%
Bomb Attacks in Iraq	10	1.24%
Russia and Chechnya	9	1.12%
Explosions in the capital Beirut	9	1.12%
IRA shooting	8	0.99%
UK deadlocked over terror bill	8	0.99%
Prison Fire in Dominican Republic	7	0.87%
Rapes in Darfur	7	0.87%
New Assembly in Iraq	6	0.74%
Pope	6	0.74%
U.S., EU Launch Joint Strategy on IRA	6	0.74%
Michael Jackson	6	0.74%
Shiites, Kurds agree on government's makeup	6	0.74%
21 die as police storm Manila jail	6	0.74%
South Korean Furor Against Japan Over Isles Rages On	6	0.74%
Insurgents attack in Iraq	5	0.62%
Terrorism in Iraq	5	0.62%
Nepal Says it will Restore Rights, Frees Prisoners	5	0.62%
3 Slain in Atlanta Courthouse Rampage	5	0.62%
Anti-Syrian Protesters Flood Lebanese Capital	5	0.62%
Air Crash in Russia	5	0.62%
Topics that had less than 5 related documents (NOT HOT TOPICS)	455	56.45%

The results presented in Table 4 show that the more the hot sentences that we cluster, the higher the CR that we can achieve. This is logical because by including more hot sentences, we should be able to cover more news stories in the corpus. However, this experiment shows that our approach makes it unnecessary to extract a large number of sentences to cover the majority of hot topics. In particular, the results show that the CR is already 100 percent for genuinely hot topics that cover 2 percent or more of all the stories in a given period of time by using only 100 top-ranked sentences.

Table 5 is the topic extraction result from using the TF*PDF algorithm, which considers only the pervasive properties of terms. The gray cells in Table 4 indicate

TABLE 4
Hot Topic Extraction Results (Pervasive and Topical)

	>5	>=1%	>=2%
SL = 30	32.35%	35.29%	50.00%
SL = 50	47.06%	52.94%	66.67%
SL = 100	64.71%	76.47%	100.00%
SL = 200	70.59%	88.24%	100.00%
SL = 300	76.47%	88.24%	100.00%

SL is the number of top-ranked hot sentences chosen to be clustered.
Column > 5 refers to topics that have more than five news reports.
Column $>= 1$ percent refers to topics that have $>= 1$ percent of all the news stories in the corpus.
Column $>= 2$ percent refers to topics that have $>= 2$ percent of all the news stories in the corpus.

combinations where our algorithm yielded equal or better results. On the surface, it may appear that the hot topic extraction results of our method is only slightly better than $\mathrm{TF}^*\mathrm{PDF}$. However, upon further investigation of the extraction results, we discovered an underlying problem with the $\mathrm{TF}^*\mathrm{PDF}$ algorithm, that is, the correctness of sentence clustering. In Table 6, we compiled the percentage of incorrect sentences that were used to identify each topic. Our results show that despite some of the high CR achieved by $\mathrm{TF}^*\mathrm{PDF}$, a large percentage of the sentences used to identify the topics, in fact, had nothing to do with the topics. These incorrect sentences belong to topics that had fewer than five documents. Thus, the CR achieved by the pure $\mathrm{TF}^*\mathrm{PDF}$ algorithm is arbitrarily high.

4.3.2 Experiment 2 (Correctness)

In Table 7, we compare the performance of our multidimensional clustering method with that of the HTV, where only 30 top-ranked hot terms are used to compare the similarity of sentences. Furthermore, we adjust the weight coefficient of each vector to determine the overall impact on the sentence clustering results. The figures on Table 7 show the percentage of sentences categorized as MS, FS, or NC when $\mathrm{SL} = 100$.

The results indicate the following points:

1. The HTV yields less accurate results because it has more misses (that is, higher MS) than our proposed multidimensional sentence vector approach.

TABLE 5
Hot Topic Extraction Results (Pervasive Only–$\mathrm{TF}^*\mathrm{PDF}$)

	>5	>=1%	>=2%
SL = 30	29.41%	41.18%	83.33%
SL = 50	32.35%	47.06%	83.33%
SL = 100	41.18%	58.82%	83.33%
SL = 200	73.53%	88.24%	83.33%
SL = 300	79.41%	88.24%	83.33%

SL is the number of top-ranked hot sentences chosen to be clustered.
Column > 5 refers to topics that have more than five news reports.
Column $>= 1$ percent refers to topics that have $>= 1$ percent of all the news stories in the corpus.
Column $>= 2$ percent refers to topics that have $>= 2$ percent of all the news stories in the corpus.

TABLE 6
Percentage of Sentences Clustered Incorrectly

	Our Algorithm	TF*PDF
SL = 30	30%	36.67%
SL = 50	24%	50.00%
SL = 100	21%	48.00%
SL = 200	17%	43.00%
SL = 300	15.67%	42.33%

2. Our approach has significantly lower misses than the HTV, but at the expense of higher values of FS and NC.
3. In all cases, the removal of the CVs resulted in performance degradation in all categories.

For point 1, let us take a closer look at one of the clusters that resulted from the HTV approach, as presented in Table 8.

In this cluster, each sentence describes different events. It appears that these sentences have been improperly clustered, even though they have the same hot term "cabinet." Although "cabinet" is a noun phrase, it is not a NE that immediately identifies a specific location or person. This explains why the MS value is high if we only use the HTV to cluster sentences.

As point 2 indicates, using multidimensional sentence vectors reduces MS significantly, albeit at the cost of increasing FS and NC. Each increase has a different explanation. For FS, increasing the dimensions of a sentence increases the risk of different NEs related to a specific topic being clustered separately. The result is that FS is higher than that derived by using the HTV method alone. On the other hand, the increase in NC correlates with the number of hot sentences being used. Table 9 shows the statistics for MS, FS, and NC when SL varies between 30 and 300. The results suggest that there is an inversely proportional relationship between SL and NC, whereas FS does not change significantly as we adjust SL. As SL increases, the number of clusters will grow, leading to a lower NC. More importantly, the low MS indicates that our proposed approach gives good quality clusters in spite of the existence of splits.

5 Conclusion

In this paper, we have proposed a system for extracting hot topics from news articles that appear in a specific time period. Our work makes two novel and important contributions:

1. a term weighting scheme that extracts genuine hot terms and
2. a multidimensional sentence modeling technique for clustering sentences.

Our proposed term weighting scheme combines two critical characteristics of hot terms, pervasiveness and topicality, to extract terms that genuinely reflect hot topics. We use the $\mathrm{TF}^*\mathrm{PDF}$ weights of terms to represent their

TABLE 7
Correctness Analysis Using $SL = 100$

Clustering Method	MS%	FS%	NC%
HTV	20%	5%	4%
No Conceptual Vectors (0.55HTV, 0.45NEV)	8%	31%	23%
No Conceptual Vectors (0.45HTV, 0.55NEV)	8%	24%	31%
No Conceptual Vectors (0.525HTV, 0.475NEV)	7%	24%	27%
No Conceptual Vectors (0.505HTV, 0.495NEV)	8%	25%	28%
All Vector (0.4HTV, 0.3NEV, 0.1DCV, 0.1PV, 0.1KV)	3%	22%	20%
All Vector (0.3HTV, 0.4NEV, 0.1DCV, 0.1PV, 0.1KV)	2%	22%	24%
All Vector (0.45HTV, 0.4NEV, 0.05DCV, 0.05PVV, 0.05KV)	3%	16%	19%
All Vector (0.49HTV, 0.48NEV, 0.01DCV, 0.01PV, 0.01KV)	3%	11%	16%

pervasive properties and apply the Aging Theory to track the life cycles of the terms over a specific period of time. The changes in the life cycle of a term determine its degree of topicality. The combination of TF*PDF and the Aging Theory overcomes the drawback of TF*PDF weighting and improves the quality of hot term extraction. In the TF*PDF scheme, TF is the dominant factor that determines whether or not a term is hot. In our proposed method, a real hot term must have a high TF*PDF value and good life cycle variation. In our experiments, most of the hot terms are noun phrases or NEs that clearly describe the roles of those involved in the events or the locations where the events occurred.

Since using keywords alone to compare the similarities between sentences is insufficient for clustering purposes, our proposed sentence modeling approach uses five kinds of sentence vectors, as opposed to one, to represent a sentence. The five vectors are the HTV, NEV, DCV, PV, and KV. The experimental results show that our approach yields substantially more accurate hot topic extraction results than traditional approaches. We validated our term weighting scheme and multidimensional sentence modeling approach on real-life news data to ensure the applicability of our work to real-world situations.

We anticipate that our work will be useful to practitioners in several fields. For example, in e-learning, our technique could be used to identify hot issues that students find difficult to understand. This kind of technology will become increasingly important as e-learning class sizes expand. Our approach may even be applied in areas related to defense and security, where hot topics may be extracted from Web sources such as blogs and wikis to uncover possible terrorist activities.

Researchers may also benefit from the results of our work in a number of ways. For example, our model could be incorporated in search engines to help researchers identify hot topics in their area of interest. This vision of having a tool that can help us discover current hot topics and their related scholarly articles is both powerful and exciting. We believe and hope that such a tool will be a reality in the not too distant future.

There are inevitably some limitations in our study as well. First, our choice of world news may be implicitly biased toward pervasive topics. It is conceivable that topics such as weather and sports are less pervasive, as they often feature news stories that are important and "hot" but only appear once per day. Further, the baseline on which we compared our extraction results was manually compiled. Hence, our baseline may be subject to human interpretation. Last, although our conceptual vectors helped improve the

TABLE 8
An Example of an HTV Cluster

Hot Term	Sentence
Cabinet	S1: A final agreement on a Syrian withdrawal will have to wait for a Cabinet to be formed and approved by Parliament.
Shiite, cabinet	S2: Shiites and Kurds then plan talks on dividing up cabinet posts in the new government.
cabinet,	S3: The island's cabinet will review economic and trade policy on China later on Monday.

TABLE 9
Percentage of MS, FS, and NC for Different SLs

	MS%	FS%	NC%
SL = 30	0.00%	10.00%	23.33%
SL= 50	2.00%	14.00%	22.00%
SL = 100	3.00%	16.00%	19.00%
SL = 200	1.50%	7.00%	15.00%
SL = 300	1.67%	9.00%	9.67%

Similarity coefficients: 0.45HTV, 0.4NEV, 0.05DCV, 0.05PV, and 0.05KV.

overall performance of hot topic extraction, we think that more vectors may be introduced to further fine tune the results. For example, some previous works have used semantics to improve the quality of hot topic extraction [11], [15]. Nallapati and Allan used sentence-level term dependencies to extract more information from individual terms [19]. Future research may take these techniques into consideration.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Council Grants 94-2217-E-002-004, 92-2416-H-002-001, and 91-2416-H-002-008.

REFERENCES

[1] A. Hotho, S. Staab, and G. Stumme, "Text Clustering Based on Background Knowledge," Technical Report 425, Inst. AIFB, Univ. of Karlsruhe, Apr. 2003.
[2] C.C. Chen, Y.T. Chen, Y. Sun, and M.C. Chen, "Life Cycle Modeling of News Events Using Aging Theory," *Proc. 14th European Conf. Machine Learning (ECML '03)*, pp. 47-59, 2003.
[3] D.M. Bikel, R.L. Schwartz, and R.M. Weischedel, "An Algorithm That Learns What's in a Name," *Machine Learning*, pp. 211-231, 1999.
[4] D.R. Cutting, D.R. Karger, J.O. Pedersen, and J.W. Tukey, "Scatter/Gather: A Cluster-Based Approach to Browsing Large Document Collections," *Proc. 15th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '92)*, 1992.
[5] G. Salton and C.S. Yang, "On the Specification of Term Values in Automatic Indexing," *J. Documentation*, pp. 351-372, 1973.
[6] G. Kumaran and J. Allan, "Text Classification and Named Entities for New Event Detection," *Proc. 27th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '04)*, pp. 297-304, 2004.
[7] H.P. Luhn, "A Statistical Approach to Mechanized Encoding and Searching of Literary Information," *IBM J. Research and Development*, vol. 1, no. 4, pp. 309-317, 1957.
[8] H.L. Chieu and Y.K. Lee, "Query Based Event Extraction Along a Timeline," *Proc. 27th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '04)*, pp. 425-432, 2004.
[9] J. Kleinberg, "Bursty and Hierarchical Structure in Streams," *Proc. Eighth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD '02)*, pp. 91-101, 2002.
[10] J. Makkonen and H. Ahonen-Myka, "Utilizing Temporal Expressions in Topic Detection and Tracking," *Proc. Seventh European Conf. Research and Advanced Technology for Digital Libraries (ECDL '03)*, 2003.
[11] J. Makkonen, H. Ahonen-Myka, and M. Salmenkivi, "Simple Semantics in Topic Detection and Tracking," *Information Retrieval*, vol. 7, no. 3–4, pp. 347-368, 2004.
[12] K. Sparck-Jones, "Index Term Weighting," *Information Storage and Retrieval*, vol. 9, no. 11, pp. 619-633, 1973.
[13] K.K. Bun and M. Ishizuka, "Topic Extraction from News Archive Using TF*PDF Algorithm," *Proc. Third Int'l Conf. Web Information Systems Eng. (WISE '02)*, pp. 73-82, 2002.
[14] M. Porter, "An Algorithm for Suffix Stripping," *Program*, vol. 14, no. 3, 1980.
[15] N. Stokes and J. Carthy, "Combining Semantic and Syntactic Document Classifiers to Improve First Story Detection," *Proc. 24th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '01)*, pp. 424-425, 2001.
[16] N. Okazaki, Y. Matsuo, N. Matsumura, and M. Ishizuka, "Activation with Refined Similarity Measure," *Proc. 16th Int'l Florida Artificial Intelligence Research Soc. Conf. (FLAIRS '03)*, pp. 407-411, 2003.
[17] R. Swan and J. Allan, "Extracting Significant Time Varying Features from Text," *Proc. Eighth Int'l Conf. Information and Knowledge Management (CIKM '99)*, pp. 38-45, 1999.
[18] R. Swan and J. Allan, "Automatic Generation of Overview Timelines," *Proc. 23rd Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '01)*, pp. 49-56, 2001.
[19] R. Nallapati and J. Allan, "Capturing Term Dependencies Using a Sentence-Tree-Based Language Model," *Proc. 11th Int'l Conf. Information and Knowledge Management (CIKM '02)*, pp. 383-390, 2002.
[20] T. Hisamitsu and J.I. Tsujii, "Measuring Term Representativeness," *Proc. 19th Int'l Conf. Computational Linguistics (COLING '02)*, vol. 1, pp. 320-326, 2002.
[21] T. Hisamitsu and Y. Niwa, "A Measure of Term Representativeness Based on the Number of Co-Occurring Salient Words," *Proc. 19th Int'l Conf. Computational Linguistics (COLING '02)*, vol. 1, pp. 1-7, 2002.
[22] TDT 2004: Annotation Manual Version 1.2, http://www.nist.gov/speech/tests/tdt/, Aug. 2004.
[23] The 2004 Topic Detection and Tracking (TDT '04) Task Definition and Evaluation Plan, http://www.nist.gov/speech/tests/tdt/, 2004.
[24] WordNet, http://www.cogsci.princeton.edu/~wn/, 2006.
[25] Y. Yang, J. Zhang, J. Carbonell, and C. Jin, "Topic-Conditioned Novelty Detection," *Proc. Eighth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD '02)*, pp. 688-693, 2002.

Kuan-Yu Chen received the BSc degree from the National Chengchi University and the MS degree in information management from National Taiwan University. He is an engineer at the Institute for Information Industry (III), Taiwan. His research interests include text mining, knowledge management, and e-commerce.

Luesak Luesukprasert received the BSc and MS degrees from Carnegie Mellon University, Pittsburgh. He is an instructor at Ming Chuan University, Taiwan. He has worked as a software developer and product manager at Dell Inc. and is now a doctoral student of information management at National Taiwan University. His current research interests include Web technologies and services, knowledge management, and data mining.

Seng-cho T. Chou received the BSc degree from the Chinese University of Hong Kong, the MS degree from the University of California, and the PhD degree in computer science from the University of Illinois, Urbana-Champaign. He is a professor of information management at National Taiwan University. His current research interests include web technologies and services, e-business and e-commerce, knowledge management, data mining, and ubiquitous and mobile computing.