2010

Modeling Anticipatory Event Transitions

He QI

Kuiyu CHANG
Nanyang Technological University

Ee Peng LIM
Singapore Management University, eplim@smu.edu.sg

DOI: https://doi.org/10.1007/978-3-540-69209-6_6

6

Modeling Anticipatory Event Transitions

Qi He, Kuiyu Chang, and Ee-Peng Lim

School of Computer Engineering,
Nanyang Technological University, Singapore

Abstract. Major world events such as terrorist attacks, natural disasters, wars, etc. typically progress through various representative stages/states in time. For example, a volcano eruption could lead to earthquakes, tsunamis, aftershocks, evacuation, rescue efforts, international relief support, rebuilding, and resettlement, etc. By analyzing various types of catastrophical and historical events, we can derive corresponding event transition models to embed useful information at each state. The knowledge embedded in these models can be extremely valuable. For instance, a transition model of the 1918-1920 flu pandemic could be used for the planning and allocation of resources to decisively respond to future occurrences of similar outbreaks such as the SARS (severe acute respiratory syndrome) incident in 2003, and a future H5N1 bird-flue pandemic. In this chapter, we study the Anticipatory Event Detection (AED) framework for modeling a general event from online news articles. We analyze each news document using a combination of features including text content, term burstiness, and date/time stamp. Machine learning techniques such as classification, clustering, and natural language understanding are applied to extract the semantics embedded in each news article. Real world events are used to illustrate the effectiveness and practicality of our approach.

6.1 Introduction

Open Source Intelligence (OSI) plays a fundamental role in Intelligence and Security Informatics (ISI), accounting for as much as 80% of the overall intelligence. In fact, former US Joint Chiefs Chairman and former Secretary of State Colin Powell said: “I preferred the Early Bird with its compendium of newspaper stories to the President's Daily Brief, the CIA’s capstone daily product”. Thus, the ability to constantly monitor and accurately track events from news sources all over the world is vital to ISI.

Major online portals like Google and Yahoo allows users to subscribe to news alerts by specifying a list of present/absent keywords to define a particular event that he or she is interested in. Unfortunately, current alert systems are not smart enough to figure out whether a news document containing all the user defined words positively actually confirms occurrence of the event. In fact, some service providers like Yahoo still entrust a human operator to approve system triggered news alerts, whereas others like Google prefer to use a completely automated approach, at the expense of generating many false alarms/alerts [9].

The *Anticipatory Event Detection* (AED) framework can uncover impending or anticipated events specified by a user. For example, it can be configured to monitor news streams for the occurrence of very specific events like “Taiwan declares independence", "Coup in Thailand", "Osama bin Laden captured", etc., which we called *anticipatory events* (AE). An AED news alert prototype has been previously reported by Chua, et al. [6].

One way to look at AED is to think of it as finding the transition between two adjacent events in an *event transition graph* (ETG). Events are represented by news articles reported before and after an *anticipatory event transition* (AET) has consummated [9, 23]. A user may only be interested in receiving a notification when a particular AET has fired, and not be bothered about the remaining AETs. If sufficient number of news articles can be collected for each of the events, it would be possible to detect any number of AETs. In order to learn a particular AET, a model will have to be trained to classify articles as occurring "before" or "after" the AET.

AED thus boils down to classifying sentences/documents into those that consume a predefined AE (hit) and those that do not. In this book chapter we present investigation of ETG modeling for AED, and also review some results on AED. The rest of this chapter is organized as follows. Sect. 6.2 surveys related work and compares AED to existing event detection tasks. In Sect. 6.3, we formally define the AED problem, types of AE detection, and subsequently propose the AED framework. We introduce various solutions for event representation suitable for AED in Sect. 6.4 and propose different classification approaches to learn the ETG in Sect. 6.5. Sect. 6.6 presents our experimental setup and results, and Sect. 6.7 concludes the chapter with a discussion of limitations and future work.

6.2 Related Work

AED falls under the broader family of problems collectively known as Topic Detection and Tracking (TDT), which hitherto includes New Event Detection (NED), Topic Tracking (TT), Retrospective Event Detection (RED), and Event Transition Graph (ETG) Modelling, etc. We shall examine each of these briefly in this section.

6.2.1 Topic and Event

The classical definition of *topic* from TDT 2004 is given as follows,

Definition 1. (Topic) *A topic is a seminal event or activity, along with all directly related events and activities.*

Note that a TDT topic has a much narrower scope than traditional IR topics or categories, and should be view more like a fine-grain news category. Likewise, a TDT *event* from TDT 2004 is defined as follows,

Definition 2. (Event) *An event refers to a particular incident occurring at a specific time and place, along with all necessary preconditions and unavoidable consequences.*

6.2.2 New Event Detection (NED)

NED, also known as *First Story Detection*, aims to detect the first story of a topic without reference to any seed news articles, i.e., it is unsupervised. The seminal paper of Allan, et al. [1] empirically showed NED to be an inherently hard problem when tackled using only cosine similarity approaches. Later studies supported this viewpoint [4, 12, 18, 22], summarized below.

Brants, et al. [4] applied a combination of techniques to NED, including Hellinger distance, tilling for better document matching, etc., and reported modestly improved NED performance.

Kumaran, et al. [12] used text classification techniques and named entities to detect *all* new events of a particular category (using a model trained threshold, i.e., supervised). They reported a mixed bag of results for different categories both with and without using named entities. For example, using solely named entities resulted in better detection rates for the legal and science categories while on the other hand, excluding named entities helped the election and sports categories. Interestingly, named entities neither help nor worsened performances for the financial category. The last observation is in agreement with our earlier findings [9], which were evaluated primarily on financial news articles. Moreover, for our AED model, we found that combining named entity types with non-named entity terms worked better than each representation alone for financial category.

Stokes, et al. [18] proposed a composite document representation using both lexical chains and proper nouns for NED, which is a more sophisticated method of applying named entities to NED.

Yang, et al. [22] reported substantial performance gain by first classifying news articles into different topics, followed by applying one nearest neighbor to detect new events (NED). This approach makes intuitive sense since a similarity comparison within news events of the same topic works better than across the board. However, along with this new approach came two new problems, namely 1) accurately classifying a news article into one or more topics, and 2) setting a reliable outlier threshold for each topic. One of the main contributions of our work on AED is the bursty document representation, which helps improve the accuracy of document classification.

All in all, despite the numerous attempts to improve NED, none have yielded spectacular results so far. This is because NED is basically an ill-posed outlier detection problem where only one class of data is known before hand, thereby suffering from the fate that a new event may be too similar to an historical event, especially within the same topic. We believe that a performance breakthrough would require a supervised approach, one that involves higher order understanding of event domain semantics. In other words, unless domain knowledge is utilized, NED will remain a hard problem for the foreseeable future.

AED is not subject to the same problems faced by NED as it simultaneously define the topic and transition type that it should monitor. Since AED is well defined as a two state problem, documents of historically similarly events can be used to train it. As such, AED is a well-posed problem and therefore theoretically much simpler than NED.

6.2.3 Topic Tracking (TT)

Topic Tracking (TT) aims to monitor and identify news articles of a specific topic based on a few training stories.

Franz and McCarley [7] formulated TT as a NED problem by replacing the document-document similarity with the similarity between a document and a cluster centroid.

Carthy, et al. [5] benchmarked two different TT systems, one keyword-based and one using lexical chaining. They used lexical chaining to discover word co-references within a sentence, and found that it significantly outperformed keyword-based systems.

There is a concept of binary state for each AET; the anticipated transition can either take place or not. In general, TT will detect and return *all* new developments of a specific topic, whereas AED will detect and return any documents reported after the specified binary transition has fired. For example, on the topic of earthquakes, TT will detect *any* new developments pertaining to a specific earthquake. In contrast, AED will fire only when a state specified by the user has been reached. For example, the firing state could be “Earthquake strikes major Chinese city with heavy casualties”. In some ways AED can be considered as a special combination of NED and TT; its topic is constrained by keywords as in TT, and it tries to detect the first story after a fired transition (albeit properly defined by training documents instead of relying on outlier threshold) just like NED.

6.2.4 Retrospective Event Detection (RED)

Closely related to AED is RED, another NED derivative, which is concerned with detecting previously unidentified events from historical news corpus [21, 13].

Yang, et al. [21] first defined the RED problem and addressed it using document clustering. Li, et al. [13] attempted to identify events within a corpus of historical date-stamped news articles with the help of both time and content information. It assumes that the news event histogram of a particular event genre is Gaussian-distributed with peaks/bursts denoting a new event. This is related to our approach of using Kleinberg's two-state automata [11] to represent term burstiness at different points in time.

The RED approach cannot be applied generally to solve the AED problem in practice since 1) it is constrained to detect generic events (such as *any* earthquake *anywhere*), and 2) it only works on historical events as it requires all (pre and post event) time information about the event in order to model it. Note that the second restriction also applies to our AED bursty model, which will be addressed in future work.

6.2.5 Event Transition Graph (ETG)

An Event Transition Graph (ETG), otherwise known as *Event Evolution Graph*, is a directed graph that models a set of events within a specific topic as nodes and edges. Specific states of the topic are represented by nodes, with the edges denoting possible transitions and associated firing conditions. Below we briefly review some previous work on ETG [14, 16, 23].

Makkonen [14] coined the term *event evolution* to denote the various time stages of a typical topic. In his work, an event is comprised of time-ordered related documents, and multiple events together constitute an event evolution graph.

Nallapati, et al. [16] defined *event threading* as the dependencies between events. They evaluated several candidate dependency graphs of clusters of news documents based on similarity, with the primary objective of analyzing the inherent structure of retrospective topics.

Yang, et al. [23] formally defined event evolution as the relationship between all events within a topic. Each relationship is in fact a transition that traverses in time from seminal events to terminal events. Their work depends heavily on the similarity metric between events, which are assumed to be comprised of well-clustered documents.

In our work, AED also assumes events to be a time-ordered sequence of documents. However, we make a simplifying assumption that a pre-existing ETG is readily available. We have yet to address the challenges of building an ETG from scratch. In principle, AED could use a combination of the above techniques to come up with a reliable ETG based on historically similar (to the current AE) events.

6.3 AED Model

6.3.1 Problem Definition

AED was originally motivated by the desire to deliver precise and customized SMS news alerts to the mobile phone subscribers [6]. As a push application with extremely high precision and recall demands, conventional keyword based alert systems simply did not cut it, and thus the birth of AED. The idea of AED is to allow a subscriber to receive only specific news alerts that he or she is interested in. A formal definition of AED is given as follows:

Definition 3. (AED) *The objective of AED is to detect and identify entities (messages or documents) that confirm the occurrence of a user specified anticipatory event transition (AET), which is also known as the user preference.*

The user preference is defined formally as follows.

Definition 4. (Anticipatory Event Transition (AET) or user preference) *A user preference or Anticipatory Event Transition (AET) corresponds to a single event transition selected from an event transition graph (ETG) of a given topic.*

Events belonging to the same topic (e.g., election of US President) often involve a common set of event transitions, e.g., nomination of party's Presidential candidates, nomination of party's Vice-Presidential candidates, election of party's Presidential team, election of Presidential team. These events collectively form an event transition graph, defined as follows.

Definition 5. (Event Transition Graph (ETG)) *An Event Transition Graph (ETG) G models multiple event transitions belonging to the same topic genre as a sequence of n events $E = [e_1, e_2, \dots, e_n]$ related by a set T of transition links between each pair of transitive events of the form*

$$T = \{t_{i,j} \mid \forall i, j \; if \; \exists transition \; e_i \; to \; e_j, where \; i < j \; and \; e_i, e_j \in E\} \qquad (6.1)$$

Thus, a user can select one amongst $|T|$ transitions as his AET or user preference.

6.3.2 AED on Document Streams

In practice, AED is usually applied to an online stream of news documents. Fig. 6.1 shows a global time-ordered sequence of documents, some on-topic and others off-topic with respect to an AET. Among the on-topic documents, only those that confirm the AET are considered hit documents, and should be identified by an AED system.

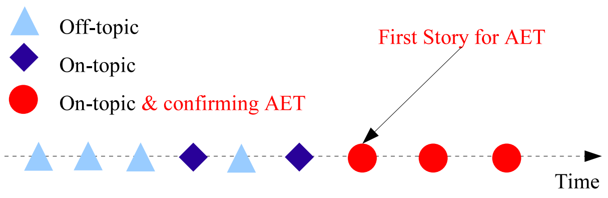


Fig. 6.1. AED for document streams

Ideally, a user should be allowed to specify any desired AET explicitly. However, this is not possible in practice due to the lack of a machine-understandable syntax to describe event semantics. One compromise is to present various known and trained ETGs to the user, from which he or she could pick a desired AET and specify the associated named entities. For example, if an ETG on the disposal of country leaders is available, a user could select from it an AET denoting resignation. The user should also supply a set of keywords such as “Taiwan President Chen Shui Bian” to indicate that he wishes to detect events confirming this particular resignation and not every resignation in the world.

Fig. 6.2 shows an ETG describing a common structure shared by all company acquisition topics. Suppose a user is interested in the event transition $t_{2,3}$ from event e_2 (“In talks to acquire”) to event e_3 (“Announces acquisition”). As multiple news articles could be associated with the “Announces acquisition” event, the earliest one will be the first story confirming the transition and is therefore the candidate document to be identified by an AED system.

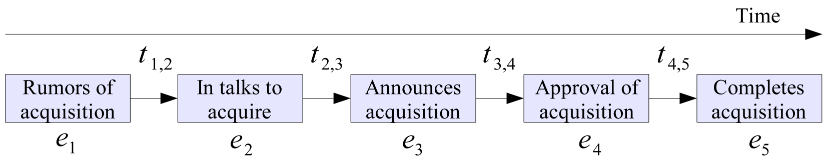


Fig. 6.2. An Event Transition Graph (ETG) for the “acquisition” topic genre

6.3.3 Types of AE Detection

Although the objective of AED is to detect the first story after the user specified AET, it may not always be successful in practice. Here, we formally define four types of AED scenarios.

Suppose we are given a set of N news articles $X = \{x_1, ..., x_N\}$ about a topic, and a sequence of n events $E = [e_1, ..., e_n]$ and its associated ETG. Each news article x_i has a publication date/time represented by $t(x_i)$ and an event type in E represented by $e(x_i)$, the latter of which is also known as the true event of x_i.

We assume that all news articles in X are sorted in time ascending order, i.e., $t(x_i) = t(x_j) \forall i < j$, and all events in E are sorted in time ascending order, i.e., $t(e_i) = t(e_j) \forall i < j$.

By applying any AE detection technique on a news article x_i, we obtained its assigned event denoted by $s'(x_i)$. Given an anticipatory event transition $t_{k-1,k}$ as the user preference, the objective of AED is therefore to find the news article x_m that satisfies:

$$x_m = \arg\min\{t(x_i) \mid \forall x_i \; where \quad s'(x_i) = e_k\} \tag{6.2}$$

To make the time comparison easier between the detected first story x_m and the event e_k, we also define the *true time* of e_k, $t(e_k)$, as follows:

$$t(e_k) = \min\{t(x_i) \mid \forall x_i \; where \quad e(x_i) = e_k\} \tag{6.3}$$

Once the first story x_m of the anticipatory event e_k is determined by the AED classifier, all subsequent news articles, x_j, $j = (m+1), \ldots, N$ will be assigned to event(s) e_k post $t_{k-1,k}$. Since the first story identified by AED may be prematured, delayed, or undefined (never found), we define four types of AED scenarios as follows:

Accurate Alarm : $t(x_m) = t(e_k)$. First story of e_k found successfully.

Delayed Alarm : $t(x_m) > t(e_k)$. First story found was too late.

False Alarm : $t(x_m) < t(e_k)$. First story found was prematured.

Miss : $t(x_m) = undefined$. No x_i in X has $s'(x_i) = e_k$. AED fails to even identify the event.

Fig. 6.3 graphically depicts each of the four types of AED scenarios. In practice, the preferred scenarios are ranked in descending order of preference as follows: accurate alarm, delayed alarm, false alarm, miss. Intuitively, a delayed alarm is preferred over a false alarm or a miss according to the age-old saying, "better late than never".

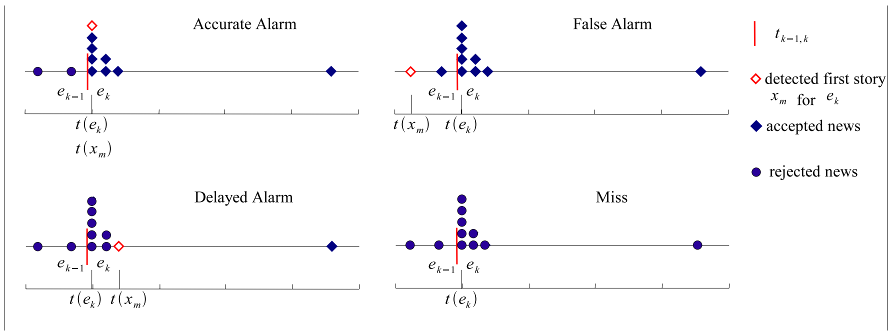


Fig. 6.3. The four AED scenarios

6.3.4 AED Prototype

Our AED prototype is shown in Fig. 6.4. Here, we make a number of simplying assumptions.

- A reliable ETG is available, from which the user selects a desired AET and enters a set of related key words (usually named entities).
- The system retrieves a set of training documents based on the selected AET. The articles could be based on historically similar events.
- The training documents are manually annotated either at the sentence or document level as belonging to one of the two possible states of the AET, pre (-) or post (+).

Based on the above assumptions, the system trains a classifier for each AET in the ETG using the labelled/annotated documents. As long as two sets of training documents for both states of a transition are available, a classifier can be pre-trained in offline mode.

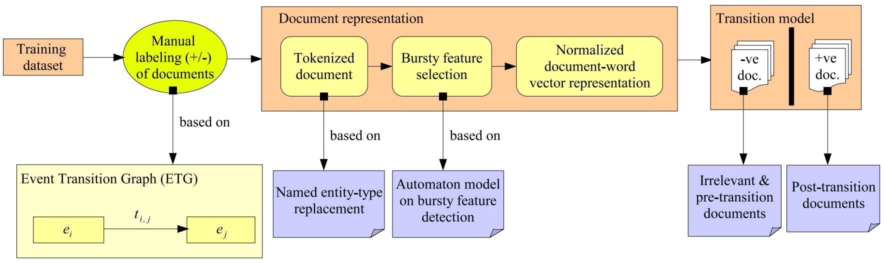

Training dataset
Manual labeling (+/-) of documents
Document representation
Tokenized document
Bursty feature selection
Normalized document-word vector representation
Transition model
-ve doc.
+ve doc.
based on
based on
based on
Event Transition Graph (ETG)
e_i
$t_{i,j}$
e_j
Named entity-type replacement
Automaton model on bursty feature detection
Irrelevant & pre-transition documents
Post-transition documents

Fig. 6.4. Online AED system showing only the selected AET of the ETG

After training, the AED prototype operates online as follows:

1. The user inputs a set of keywords describing the desired event and selects the appropriate AET from an available ETG.
2. The system monitors an online news stream and filters off a set of candidate documents matching the user specified keywords.
3. The trained classifier corresponding to the user selected AET is applied to this candidate set, where each document is classified as negative or positive. Once a positive document is found, the AED system is deemed to have detected the anticipatory event.

6.4 Document Representation for AED

An essential portion of our AED system lies in the document representation format, as shown in Fig. 6.4. In this section, we describe various approaches to effectively represent event semantics, which can lead to better AED results.

6.4.1 Extracting Named Entities Types from Documents

In order to train a classifier on an AET using historically similar documents, it is very important to have the named entities replaced by named entity types [9]. For example, consider the following statements referring to two different "announces acquisition" events, with the named entities in boldface:

"China's biggest computer maker, ***Lenovo Group***, said on ***Wednesday*** it has acquired a majority stake in ***IBM Corp***'s personal computer business in a deal worth a total value of ***US$1.75 billion*** (***S$2.86 billion***), one of the biggest ***Chinese*** overseas acquisitions ever."

"***SBC Communications*** on ***Monday*** announced plans to acquire ***AT&T*** in a ***$16 billion*** deal, a move designed to bolster ***SBC***'s sales to enterprise customers nationwide and give it new national and global networks."

In order for the two statements to be representative of the "post" state of the transition, it is better to replace the named entities with their name entity types, as follows:

"***GPE***'s biggest computer maker, ***ORGANIZATION***, said on ***DATE*** it has acquired a majority stake in ***ORGANIZATION***'s personal computer business in a deal worth a total value of ***MONEY*** (***ORGANIZATION MONEY***), one of the biggest ***NATIONALITY*** overseas acquisitions ever".

"***ORGANIZATION*** on ***DATE*** announced plans to acquire ***ORGANIZATION*** in a ***MONEY*** deal, a move designed to bolster ***ORGANIZATION***'s sales to enterprise customers nationwide and give it new national and global networks."

Clearly, after the replacement, the two examples become more similar to one other, which invariably helps the classifier learn the event transition better.

6.4.2 Factoring Burstiness into Document Representation

Motivation for Bursty Feature Representation

An up and coming topic is usually accompanied by a sharp rise in the reporting frequency of some distinctive features, known as "bursty features". These bursty features could be used to more accurately portray the semantics of an evolving topic. Fig. 6.5 illustrates the effectiveness of using top bursty features to represent two separate topics. Had we used the usual feature selection and weighting scheme, the word features "Gingrich" and "Newt" frequent in both related but different topics would turn up nearly important for representing documents of these two topics.

Thus, the classical static Vector Space Model (VSM) [17] simply is not ideal in representing evolving trends in text streams, nor is it able to meaningfully model a transition from one semantic context to another. We therefore propose a new text

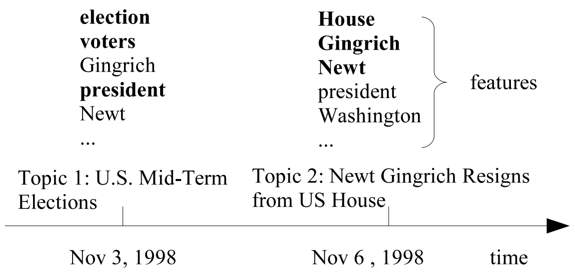


Fig. 6.5. Frequent features of two topics (bursty features shown in bold)

stream representation model, called bursty feature representation, which can emulate sophisticated temporal and topical behaviour via bursty features, as illustrated in Fig. 6.5. In our model, a burst corresponds to the definition as follows

Definition 6. (burst) *A burst is a phenomenon in which a large amount of text content about the same topic is generated in a short time period.*

Bursty Topic Representation

A bursty topic can be identified by modeling the document frequency (DF) with Kleinberg's two state automaton [11]. Likewise, the DF of every word can also be modeled, thereby uncovering bursty words. Fig. 6.6 shows an example of a bursty topic "Clinton's Gaza Trip" from the TDT3 dataset.

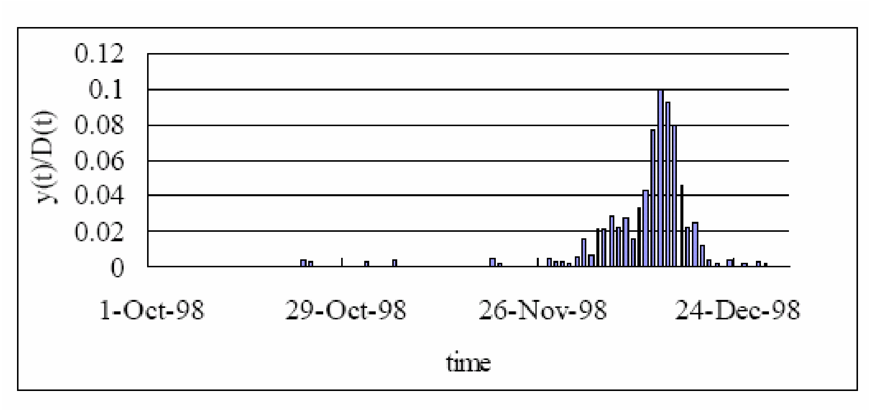


Fig. 6.6. A bursty topic (Clinton's Gaza Trip) taken from TDT3, shown as a plot of the fraction of on-topic document frequency versus time.

Going a step further, we proposed representing a document with bursty features [10], which involves two steps: (1) identifying bursty features, and (2) representing documents using bursty features/weights.

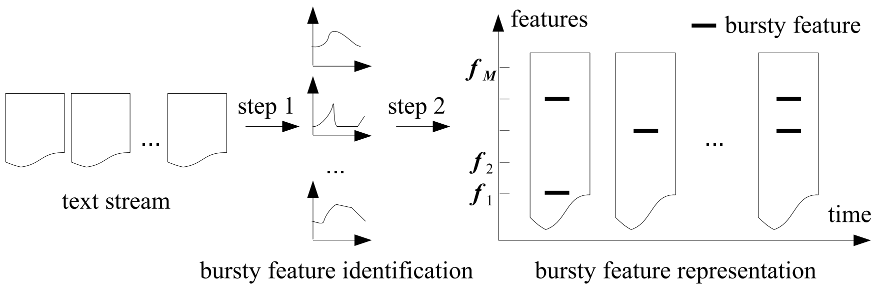


Fig. 6.7. An overview of bursty feature representation

Fig. 6.7 shows how a document is assigned bursty weights that are dependant on its time stamp *t*. The same raw document may have different bursty feature representations at two different time points $t_i \neq t_j$.

We now formally describe our bursty feature representation that combines burstiness with static feature weights [10]. Let $F = \{f_j; j = 1, 2, \cdots, M\}$ be the static VSM feature space, and FP_{ij} indicates the static feature weight (i.e., binary weighting) of f_j in document $\mathbf{d}_i$. Let *B* represent the bursty feature space where $B \subseteq F$.

Definition 7. (Bursty Feature Representation) *A document* $\mathbf{d}_i(t)$ *at time t has a bursty feature representation in the form*

$$\mathbf{d}_i(t) = [d_{i1}(t), d_{i2}(t), \cdots, d_{iM}(t)]^T \tag{6.4}$$

where

$$d_{ij}(t) = \begin{cases} FP_{ij} + \delta w_j & \text{if } f_j \in B \text{ and } t \in p_j \\ FP_{ij} & \text{otherwise} \end{cases} \tag{6.5}$$

where $\delta > 0$ *is the burst coefficient,* w_j *is the bursty weight of a bursty feature* f_j *and* p_i *is the bursty period of* f_i.

Here, the role of δ is to combine the sufficiency of the static VSM feature space with the discriminatory properties of bursty features. In other words, bursty features are enhanced or boosted by a factor of δw_j, whereas non-bursty documents will simply fall back to their static feature representation as explained in Fig. 6.8.

Under the general assumption that bursty features are representative and unique for each topic, we showed theoretically that the bursty feature representation will always improve the objective function of a clustering solution [10].

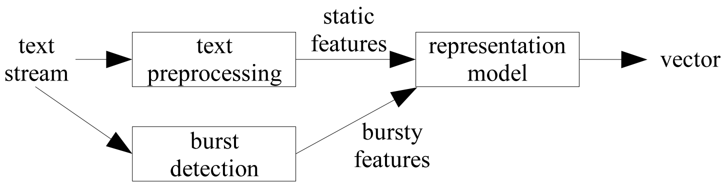


Fig. 6.8. Bursty feature representation

6.5 Modeling the AET

We have experimented with two different resolutions for training the transition model of Fig. 6.4, 1) sentence resolution, and 2) document resolution. Training at the sentence resolution is not only hard, but also extremely labour intensive because every sentence has to be manually annotated. In this section, we describe both approaches.

6.5.1 Sentence Classifier

A sentence classification model was initially built to model the AET [8]. We considered the sentence resolution based on the intuition that the most representative sentences from on-topic AET confirming articles typically provide a good summary of the transpired event transition.

In general, an event transition can be confirmed from a sentence, given enough contexts. For example, the following sentence would qualify as a "hit" sentence for the anticipatory event "win basketball match".

1. Hit Sentence: "*The Knicks outscored Philadelphia 32-22 in the fourth quarter to secure the win.*"
2. User Preference (AET): "*win basketball match.* "

Single-Level and Two-Level SVM Sentence Classifiers

For the sentence classification model, we proposed a simple single-level support vector machine (SVM) sentence classifier and a more sophisticated two-level hierarchical SVM sentence classifier.

The single-level SVM sentence classifier simply classifies all sentences as either positive (i.e., on-topic and event confirming) or negative (i.e., on-topic but non-event confirming, and off-topic).

The two-level SVM classifier attempts to distinguish sentences about current events from those about historical events as these sentences could otherwise confuse the single-level sentence classifier that is also responsible for distinguishing on- and off-topic sentences. The sentences about current events and historical events are known as *positive* and *historical* sentences respectively. For example, "the rejuvenated Celtics have won three straight since then and six straight at home overall" is a typical historical sentence, which is considered as "on topic" by the single-level classifier but hard to identified as non-event confirming because the single level classifier is not trained to distinguish event confirming sentences from non-event confirming sentences. After applying two-level classification, this confusing case is easily solved.

Fig. 6.9 shows the structure of the two-level SVM classifier. The first level classifier aims to detect all on-topic sentences, which include both positive and historical

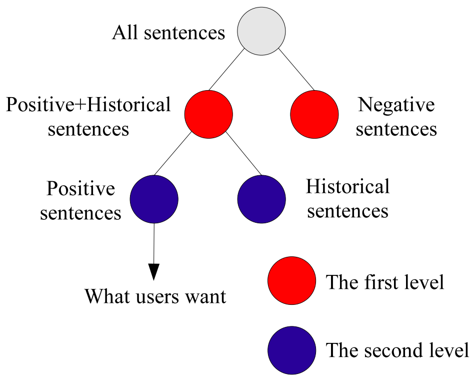


Fig. 6.9. 2-level SVM sentence classifier

sentences. The second level classifier performs a refinement on the on-topic sentences by further classifying them as positive or historical.

Sentence Classification Methods
We investigate various sentence retrieval strategies for AED, with a substantial focus on improving retrieval quality. In practice, the term weighting scheme used to represent a sentence vector has an enormous impact on the classification accuracy. The following methods using different term weighting schemes were compared in our experiments:

- Single-Level Classifier with {Standard TF, TFIDF, TFISF, TF+named entity features}
- Two-Level Classifier with {Standard TF, TFIDF, TFISF, TF+named entity features}

The standard TF scheme simply uses the raw frequency count of each term within a sentence. Another important factor to consider is the distribution of terms across a collection. Usually terms that are limited to a few sentences are useful for discriminating those sentences from the rest of the collection. This assumption leads to the introduction of ISF, called *inverse sentence frequency*. We also introduced the IDF, called *inverse document frequency*, at the sentence level to assume that terms appearing in a small number of documents are useful. The various term weighting schemes are summarized as follows:

$$\mathit{Standard\ TF}: f_{ij} \tag{6.6}$$

$$\mathit{TFIDF}:\ f_{ij} \times \log(\frac{N}{n_i}) \tag{6.7}$$

$$\mathit{IFISF}:\ f_{ij} \times \log(\frac{S}{s_i}) \tag{6.8}$$

where f_{ij} is the frequency of term i in sentence j, N is the total number of documents in the collection, S is the total number of sentences in the collection, n_i is the number of documents containing term i, and s_i is the number of sentences containing term i. Our proposed weighting scheme, TF with named entities is simply standard TF appended with some domain named entity features denoting the frequencies of them.

6.5.2 Document Classifier

Sentence retrieval is a very difficult problem [2] by itself. This is because a single sentence contains neither enough information (curse-of-dimensionality) nor context to form a meaningful model. Thus, we proposed modeling the AET transition at the document resolution [9].

Document Classification Methods
We tried three different feature representation methods and one classifier combining strategy to train the AET classifier, as follows:

CONTENT	: Entire news content as features.
TITLE	: Title as features.
1SENT	: First sentence as features.
VOTING	: Majority voting on the above three classifier outputs.

The TITLE and 1SENT representations were inspired by the observation that human experts can usually decide if a news is a hit simply based on its first sentence and/or title. Moreover, the TITLE and 1SENT representation of a news article may not always carry useful features, and the AED decision will have to fall back to the CONTENT representation. For example, the first sentence "*Signature Control Systems is off to a busy start in early 2006*" does not contain features really relevant to the "*acquisition*" event transition. VOTING was thus used as a simple and effective way to improve the overall accuracy.

6.6 Experiments

6.6.1 Dataset

Since AED is a relatively new area of research, we created three customized datasets, namely *basket100* to test the sentence AED model, *Google Acquisition* and *Acquisition7* to test the document AED model for the "mergers and acquisitions" topic genre.

To evaluate our bursty document representation, we created the *TDT3-Eng* set, which is comprised of documents from 116 topics extracted from the TDT3 collection.

TDT3-Eng Dataset

The TDT3 dataset includes 51,183 news articles collected during the 3 month period of October through December 1998. Among these, 37,526 English articles originated from 8 English sources, and 13,657 Chinese articles came from 3 Chinese sources. We extracted a subset of 8,458 on-topic English news articles covering 116 topics as *TDT3-Eng*.

After stop-word removal, 125,468 distinct features remained in *TDT3-Eng*. Among these, 2,646 were identified as bursty (set *B*) using the 2-state automaton model described in [10]. We independently selected another 2,646 features (set *F*) using the document frequency thresholding technique [20].

For a fair comparison, only bursty features in $F \cap B$ are used in our bursty feature representation. Finally we have 1,394 distinct bursty features ($|F \cap B| = 1{,}394$) with 1,863 bursts, averaging 1.34 bursts per bursty feature.

Basket100 Dataset

The *Basket100* collection comprises 100 documents returned by Google using the user preference "win basketball match". In *Basket100*, 93 out of 100 documents are relevant, i.e., describes basketball games, and the remaining 7 are irrelevant. The collection contains 2,340 sentences, compris ing 4,499 unique terms (words). The 2,340 sentences were manually annotated into 3 categories:

1. positive-current class for “current basketball result”
2. negative-historical class for “historical basketball results”
3. negative class for “irrelevant” or off-topic sentences

Table 6.1 shows the summary statistics for *Basket100.*

Table 6.1. Class distribution of Basket100

Classes	*Count*
Positive documents (class 1: win basketball event)	93
Negative documents (class 2: irrelevant)	7
Total	100
Positive sentences (class 1: current win basketball event)	189
Negative sentences (class 2: historical win basketball event)	117
Negative sentences (class 3: other irrelevant sentences)	2,034
Total	2,340

Google Acquisition Dataset

We would like to find a way of automatically retrieving the training dataset for event transition detection. Therefore, *Google Acquisition* dataset, which contains 346 as-it-happens news articles returned by Google News Alerts during the two-month period from Dec 19, 2005 to Feb 19, 2006 using the user preference “announces acquisition”, is created.

In *Google Acquisition*, 178 documents were labelled as positive and 168 as negative w.r.t the “announces acquisition” transition, which means that Google News Alerts returned 168 (48.6%) outright false alarms for the subscribed keywords “announces acquisition”. This is a typical result from a simplistic keyword-based news alert system.

Acquisition7 dataset

Another dataset, *Acquisition7*, which covers seven recent acquisition topics, was created as the test data for the document classification model. Each acquisition news topic in *Acquisition7* is comprised of 20 news articles returned by Google News, approximately half of each (10) were reported before and after “announces acquisition” transition.

The 7 acquisition news topics are listed in Table 6.2, where *t(e)* refers to the true occurrence date for “announces acquisition” event.

6.6.2 Experimental Setup

Version 2 of the open source Lucene software was used to tokenize the news text content, remove stop words, and generate the document-word vector. In order to preserve time-sensitive past/present/future tenses of verbs, no stemming was done other than the removal of a few articles. SVM Cost factors [15] were used to deal with the highly unbalanced class sizes.

Table 6.2. Make up of the *Acquisition7* dataset.

Acquisition Topics	$t(e)$
Adobe acquires Macromedia	Apr 18, 2005
CNPC acquires PetroKazakhstan	Oct 26, 2005
eBay acquires Skype	Sep 12, 2005
Lenovo acquires IBM PC Division	Dec 08, 2004
Oracle acquires PeopleSoft	Dec 13, 2004
Oracle acquires Siebel	Sep 12, 2005
SBC acquires AT&T	Jan 31, 2005

In our experiments, we use BBN's Identifinder [3] to identify 24 types of named entities, including *Animal*, *Contact info*, *Disease*, *Event*, *Facility*, *Game*, *Geo-political entities*, *Language*, *Law*, *Location*, *Nationality*, *Organization*, *Person*, *Plant*, *Product*, *Substance*, *Work of art*, *Date*, *Time*, *Cardinal*, *Money*, *Ordinal*, *Percentages*, and *Quantity*. Extracted named entities are then replaced in line by one of the 24 named entity types.

6.6.3 Clustering *TDT3-Eng*

We applied K-means (K = 116) clustering to **TDT3-Eng**, which comprises 116 topics or classes. Since bursty features are identified based on *TDT3-Eng* itself, the bursty coefficient δ is set to 1 as suggested in [10].

Assume that K clusters are generated. Let $|k_j|_{C_i}$ denote the number of documents from topic C_i assigned to cluster k_j. Similarly, let $|C_i|_{k_j}$ denote the number of documents from cluster k_j originating from class C_i.

We evaluated our clustering results using three standard metrics: cluster purity, cluster entropy, and class entropy defined as follows:

Definition 8. (Purity) *The purity of cluster k_j is defined by*

$$purity(k_j) = \frac{1}{|k_j|}\max_i(|k_j|_{C_i}) \tag{6.9}$$

The overall purity of a clustering solution is expressed as a weighted sum of individual cluster purities

$$cluster\ purity = \sum_{j=1}^{K}\frac{|k_j|}{|D|}purity(k_j) = \frac{1}{|D|}\sum_{j=1}^{K}\max_i |k_j|_{C_i} \tag{6.10}$$

Definition 9. (Cluster Entropy) *Cluster entropy measures the diversity of a cluster k_j, and is defined as*

$$entropy(k_j) = -\sum_i \frac{|k_j|_{C_i}}{|k_j|} \log \frac{|k_j|_{C_i}}{|k_j|} \tag{6.11}$$

The total entropy of a cluster solution is

$$cluster\ entropy = \sum_{j=1}^{K} \frac{|k_j|}{|D|} entropy(k_j) \tag{6.12}$$

Both cluster purity and entropy measure the homogeneity of a cluster, but neither of them measures the recall of each topic. Thus, we introduce class entropy as follows

Definition 10. *(Class Entropy) The class entropy of a cluster is defined as:*

$$entropy(C_i) = -\sum_j \frac{|C_i|_{k_j}}{|C_i|} \log \frac{|C_i|_{k_j}}{|C_i|} \tag{6.13}$$

The total class entropy of a cluster solution is

$$class\ entropy = \sum_{i=1}^{K} \frac{|C_i|}{|D|} entropy(C_i) \tag{6.14}$$

In general, a good clustering algorithm should have high cluster purity, low cluster entropy, and low class entropy.

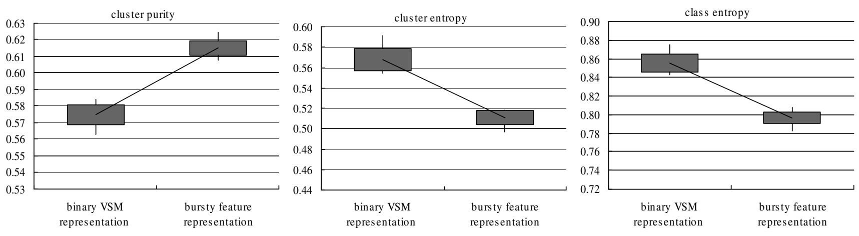


Fig. 6.10. Averaged clustering results for *TDT3-Eng* over 10 runs, showing the mean (end points of line joining the two box plots), spread (box), and range (vertical line).

Table 6.3 lists the 3 evaluation metrics averaged over 10 clustering runs for the binary VSM and bursty feature representations. The metrics are also plotted in Fig. 6.10, which shows the mean, spread (standard deviation) in each direction, and range.

Table 6.3. Averaged clustering results for *TDT3-Eng* over 10 runs

representation	*cluster purity*	*cluster entropy*	*class entropy*
binary VSM	0.5750	0.5682	0.8553
bursty feature	**0.6149**	**0.5110**	**0.7971**
Improvement	**6.93%**	**10.06%**	**6.81%**

From Table 6.3, we see that bursty feature produces clusters with on average 10.06% and 6.81% lower cluster and class entropies, respectively, and 6.93% higher cluster purity. Fig. 6.10 further highlights that bursty feature representation yields more consistent and stable clustering solutions with lower variance and smaller range in the three metrics.

The results are very encouraging considering that 1) many of the topics in *TDT3-Eng* are small (with just a few documents) and non-bursty, and 2) there is a fair amount of overlap in bursty feature space between the various topics, which clearly violated the non-overlapping assumption.

6.6.4 Modeling Transitions at the Sentence Resolution

Two classifiers using various term weighting schemes of the sentence model were applied to the *Basket100* dataset, with the goal of detecting a winning basketball event. We evaluate the sentence classification performance using the standard precision, recall and F-Measure metrics defined as:

$$Precision = \frac{\#correct\ postitive\ predictions}{\#positive\ predictions} \tag{6.15}$$

$$Recall = \frac{\#correct\ postitive\ predictions}{\#positive\ samples} \tag{6.16}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision \times Recall} \tag{6.17}$$

Single-level SVM Sentence Classifier

Fig. 6.11 shows the classification results of the single-level SVM. We see that the sentence classifier using our proposed weighting scheme yielded the best F-Measure of 0.69, leading the next competitor by 15%. Moreover, the recall of 0.63 is low by practical standards, despite it beating the nearest competitor (TF) by more than 20%.

The other methods fared significantly worse. TFISF performed worse than TF, probably due to the fact that there were too many negative (including historical) sentences, thereby distorting the ISF. Note that for single-level classification, the positive and historical winnings are labelled differently, despite them sharing a common vocabulary, e.g., “win”, “loss”, etc. TFIDF performed the worst, due to the large discrepancies between the importance of a term at the sentence and document level.

Two-level SVM Sentence Classifier

Fig. 6.12 shows the results of the two-level classifier. Since the first level classifier is only responsible for distinguishing on-topic sentences from off-topic ones, its performance was measured based on all on-topic sentences which included historical sentences.

Figs. 6.12(a)–(b) show that the precision values at both levels were not affected much by the different weighting schemes, unlike with the single-level classifier. This

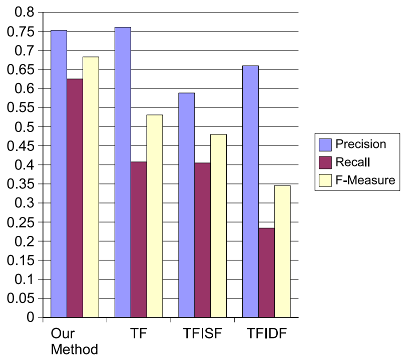
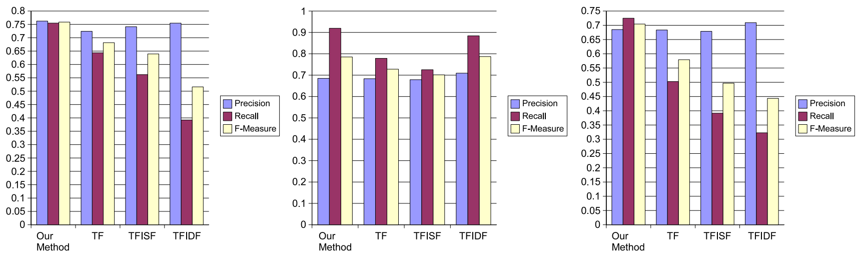


Fig. 6.11. Cross validated (10-fold) results of single-level SVM classifier

confirmed our previous suspicion that the similarity between positive and historical sentences was a large contributing factor to the low precision when inverse document and sentence frequencies come into play for the single-level classifier. The overall performances of the two-level classifier is shown in Fig. 6.12(c), with our method achieving the overall best result of 0.69 precision and 0.72 recall.



(a) Test accuracy at first level. (b) Test accuracy at second level. (c) Overall test accuracy.

Fig. 6.12. Cross validated (10-fold) results of two-level SVM classifier

6.6.5 Modeling Transitions at the Document Resolution

Before we test our trained transition model, we need to evaluate its raw cross-validated performance, to make sure it is a decent model. Afterwhich, we evaluated its AET performance on a given set of unseen AE by tallying the total number of *false alarms*, *delayed alarms*, *accurate alarms*, and *misses*.

Validating Google Acquisition Dataset

In order to validate the generic "announces acquisition" trained model, we conducted two-fold cross-validated experiments using the four text classification approaches of Sect. 6.5.2 on the *Google Acquisition* dataset. The dataset is first split along the time-line into two equal parts: 1) news articles dating from Dec 19, 2005 to Jan 19, 2006,

Table 6.4. Average test results on *Google Acquisition*. Best results are shown in bold.

Average	*CONTENT*	*TITLE*	*1SEN*	*VOTING*
False Alarms	22.5	15.5	17	**13.5**
Misses	**9**	24.5	15	10
Precision	0.7847	0.8110	0.8172	**0.8571**
Recall	**0.9011**	0.7308	0.8352	0.8901
F1	0.8389	0.7688	0.8261	**0.8733**

and 2) news articles dating from Jan 20, 2006 to Feb 19, 2006. One part was used for training with the other part used for testing and vice-versa.

The significance of this experiment shown in Table 6.4 is that it increased the precision of Google's returned news alerts from 51.4% to 85.7%, a more than 33% improvement! Furthermore, the high precision and recall figures confirmed that the *Google Acquisition* dataset is indeed suitable for modelling the transition into the "announces acquisition" event.

Testing the AET Model on Acquisition7 dataset

In this section, we test the generic AED classifier trained by *Google Acquisition* on the *Acquisition7* dataset. One AED outcome is shown in Fig. 6.13. Note that once the "first" story of "announces acquisition" event has been identified by AED, all subsequent news articles are labelled "positive".

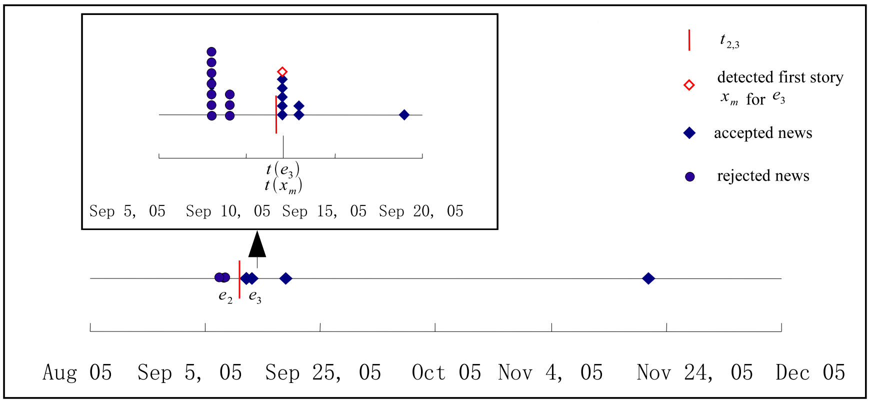


Fig. 6.13. Online AED of "eBay acquires Skype" found an accurate alarm, $t(x_m) = t(e_3)$. *transition*$_{2,3}$ is the "announces acquisition" transition in Fig. 6.2.

Table 6.5 gives a summary of the overall performances, which shows that AED based on the VOTING method generated 4 accurate alarms, 1 delayed alarm, 2 false alarms, and 0 misses. This means that the model trained by *Google Acquisition* was able to cover the main characteristics of all 7 acquisition topics. The results shown here is markedly better than methods based on cosine similarity, which failed for all except one of the 7 events [9].

Table 6.5. AED results on *Acquisition7* using the VOTING method

Alarms:	*Accurate*	*Delayed*	*False*	*Miss*
Adobe acquires Macromedia	√			
CNPC acquires PetroKazakhstan	√			
eBay acquires Skype	√			
Lenova acquires IBM PC Division			√	
Oracle acquires PeopleSoft			√	
Oracle acquires Siebel	√			
SBC acquires AT&T		√		

6.7 Conclusions and Future Work

6.7.1 Conclusions

We proposed a new practical application called Anticipatory Event Detection (AED), which is a more refined and personalized form of event tracking and detection. We then investigated suitable document presentation and various classification methods to tackle the AED problem, with a substantial focus on improving the AED transition models, which were verified experimentally on restricted domains.

AED has an essential application in ISI news alerts system, which can help information analysts to monitor only relevant events. The holy grail of AED is to detect any number of AE transitions of arbitrary genres. This is akin to having a live assistant constantly scanning newsfeed monitoring a set of AEs. Our current contributions have achieved the goal of verifying the feasibility of training AE transitional models for homogenous future events, and investigated the burstiness nature of representative features for important events by improving the traditional IR document representation.

The main limitation of AED lies in its reliance on a pre-trained transition model for every user-specified anticipatory event. This means that in practice, a user is not allowed to specify any anticipatory event, but instead must choose from a list of available pre-trained anticipatory event transitions and ETGs, e.g., terrorist bombing, earthquake disaster, mergers and acquisitions, sports scores, etc. The flip-side of this is that accurate ETG can be built for topics that matters the most to ISI analysts, and extremely accurate detection rates can be achieved.

6.7.2 Future Work

For the foreseeable future, we envisage a real-time feedback AED system as a testbed for conducting experiments on different AED methods, by allowing a subscriber to refine his/her anticipatory event definition using similar historical events. For example, to define an anticipatory event such as “China attacks Taiwan”, the user can specify a similar transpired event like “Iraq invades Kuwait”, and manually supply the set of “pre” and “post” documents of the historical event, from which the AED system can learn the transition.

Secondly, we would like to introduce outlier detection for an event transition along with more sophisticated distribution of high-dimensional free text, semi-automatically build ETGs for different topic types by applying clustering, and further investigate the burstiness properties of important events. With the above improvements, the AED system could very well become a truly reliable and personalized alert system that anyone can put to good practical use.

References

1. Allan, J., Lavrenko, V., Jin, H.: First story detection in TDT is hard. In: CIKM 2000, pp. 374–381 (2000)
2. Allan, J., Wade, C., Bolivar, A.: Retrieval and Novelty Detection at the Sentence Level. In: SIGIR 2003, pp. 314–321 (2003)
3. Bikel, D.M., Schwartz, R., Weischedel, R.M.: An algorithm that learns what's in a name. Machine Learning 34(1-3), 211–231 (1999)
4. Brants, T., Chen, F., Farahat, A.: A system for New Event Detection. In: SIGIR 2003, pp. 330–337 (2003)
5. Carthy, J.: Lexical Chains for Topic Tracking. PhD thesis, Department of Com-puter Science, National University of Dublin (2002)
6. Chua, K., Ong, W.S., He, Q., Chang, K., Kek, A.: Intelligent Portal for Event-triggered SMS Alerts. In: IEE Mobility (2005)
7. Franz, M., Ward, T., McCarley, J.S., Zhu, W.J.: Unsupervised and supervised clustering for topic tracking. In: SIGIR 2001, pp. 310–317 (2001)
8. He, Q., Chang, K., Lim, E.P.: Anticipatory Event Detection via Sentence Classification. In: IEEE SMC 2006, pp. 1143–1148 (2006)
9. He, Q., Chang, K., Lim, E.P.: A Model for Anticipatory Event Detection. In: Embley, D.W., Olivé, A., Ram, S. (eds.) ER 2006. LNCS, vol. 4215, pp. 168–181. Springer, Heidelberg (2006)
10. He, Q., Chang, K., Lim, E.P.: Bursty Feature Representation for Clustering Text Streams. In: SIAM Data Mining 2007 (2007)
11. Kleinberg, J.: Bursty and Hierarchical structure in streams. In: SIGKDD 2002, pp. 91–101 (2002)
12. Kumaran, G., Allan, J.: Text classification and named entities for new event detection. In: SIGIR 2004, pp. 297–304 (2004)
13. Li, Z.W., Wang, B., Li, M.J., Ma, W.Y.: A probabilistic model for retrospective news event detection. In: SIGIR 2005, pp. 106–113 (2005)
14. Makkonen, J.: Investigations on Event Evolution in TDT. In: HLT-NAACL 2003, pp. 43–48 (2003)
15. Morik, K., Brockhausen, P., Joachimss, T.: Combining statistical learning with a knowledge-based approach – a case study in intensive care monitoring. In: ICML 1999, pp. 268–277 (1999)
16. Nallapati, R., Feng, A., Peng, F., Allan, J.: Event Threading within News Topics. In: CIKM 2004, pp. 446–453 (2004)
17. Salton, G., Buckley, C.: Term-weighting Approaches in Automatic Text Retrieval. Information Processing and Management 24, 513–523 (1988)
18. Stokes, N., Carthy, J.: Combining semantic and syntactic document classifiers to improve first story detection. In: SIGIR 2001, pp. 424–425 (2001)
19. TDT04, TDT: Annotation Manual Version 1.2, August 4 (2004), `http://www.ldc.upenn.edu/Projects/TDT2004`

20. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: ICML 1997, pp. 412–420 (1997)
21. Yang, Y., Pierce, T., Carbonell, J.: A study on retrospective and on-line event detection. In: SIGIR 1998, pp. 28–36 (1998)
22. Yang, Y., Zhang, J., Carbonell, J., Jin, C.: Topic-conditioned Novelty Detection. In: SIGKDD 2002, pp. 688–693 (2002)
23. Yang, C.C., Shi, X.D.: Discovering event evolution graphs from newswires. In: WWW 2006, pp. 945–946 (2006)

Online Resources

1. Open Source Intelligence in Wikipedia:
   http://en.wikipedia.org/wiki/open_source_intelligence
2. Google News with News Alerts subscription:
   http://news.google.com
3. TDT: Topic Detection and Tracking Research:
   http://www.nist.gov/speech/tests/tdt/index.htm
4. TDT 2004: Annotation Manual Version 1.2, August 4 2004:
   http://www.ldc.upenn.edu/Projects/TDT2004
5. TDT3: Topic Detection and Tracking Dataset 3:
   http://projects.ldc.upenn.edu/TDT3
6. Support Vector Machines for classifying text:
   http://svmlight.joachims.org
7. Apache Lucene-Core 2.0.0: http://lucene.apache.org

Questions for Discussions

1. Does text summarization help AED?
2. Is it possible to represent a document-word vector using only bursty features for AED? If yes, how? If no, why?
3. What is the major difference between AED and an event driven search engine?
4. Why do we need to train the transition model for an AE? Could we simply use online unsupervised clustering?
5. How can we automatically train a ETG for certain topic genres? Does similarity comparison support multiple outcomes from one seminal event?
6. How can we statistically select a threshold to differentiate bursty feature from normal features?
7. (Scenario study) The CIA would like to monitor an underground extremist run discussion forum for any new terrorist attack plans. What should be the first step if CIA deploys an AED system for this?