EventSearch: a system for event discovery and retrieval on multi-type historical data

Article · August 2012

DOI: 10.1145/2339530.2339781

Dongdong Shan
Peking University
8 PUBLICATIONS **194** CITATIONS
SEE PROFILE

Ziqi Wang
Peking University
12 PUBLICATIONS **93** CITATIONS
SEE PROFILE

Junjie Yao
University of Glasgow
20 PUBLICATIONS **568** CITATIONS
SEE PROFILE

Hongfei Yan
Peking University
37 PUBLICATIONS **1,399** CITATIONS
SEE PROFILE

EventSearch: A System for Event Discovery and Retrieval on Multi-Type Historical Data

Dongdong Shan+ Wayne Xin Zhao+ Rishan Chen+ Baihan Shu+ Ziqi Wang++ Junjie Yao++
Hongfei Yan+* Xiaoming Li++
School of Electronics Engineering and Computer Science, Peking University, Beijing, China
{sdd, zhaoxin, crs, sbh, yhf }+@net.pku.edu.cn {junjie.yao, wangziqi, lxm}++@pku.edu.cn

ABSTRACT

We present EventSearch, a system for event extraction and retrieval on four types of news-related historical data, i.e., Web news articles, newspapers, TV news program, and micro-blog short messages. The system incorporates over 11 million web pages extracted from "Web InfoMall", the Chinese Web Archive since 2001. The newspaper and TV news video clips also span from 2001 to 2011. The system, upon a user query, returns a list of event snippets from multiple data sources. A novel burst model is used to discover events from time-stamped texts. In addition to offline event extraction, our system also provides online event extraction to further meet the user needs. EventSearch provides meaningful analytics that synthesize an accurate description of events. Users interact with the system by ranking the identified events using different criteria (scale, recency and relevance) and submitting their own information needs in different input fields.

Categories and Subject Descriptors
H.3.3 [**INFORMATION STORAGE AND RETRIEVAL**]: Information Search and Retrieval.

General Terms
Algorithms, Performance.

Keywords
event detection, event search.

1. INTRODUCTION

Along with the rapid growth of the World Wide Web, various news-related Web data is generated in a dramatic rate, and spreads throughout the Internet. Not only limited to traditional text based news, on-line news data usually contains various types of information, including news articles, news videos and micro-blog short messages. In order to facilitate the process of accessing news and revisiting retrospective events, it is essential to analyze and mine the information contained in these data. Thus, there is an emergent need of a comprehensive event extraction and retrieval system, which can help users to manage the overflow of news information and to extract valuable knowledge from on-line news sources.

A significant number of approaches to event mining and news exploration systems have been reported recently[1,4,5]. The most prevalent way is to detect events by grouping topically similar news articles into clusters [5]. However, most of previous studies focus on a single type of textual news data, e.g., either on news articles [1,5] or tweets [4]. Due to the complex nature of news information, a single stream, which only considers textual information, may not be able to form a comprehensive view of events. News- related data often involve various types of information, and it is beneficial to combine different types of information for event discovery and visualization. In fact, mainstream newspaper websites, e.g., New York Times, have extended the traditional text based news by incorporating multimedia information, e.g., videos and photos. Such a mechanism significantly improves user experience and utilities when reading news. In addition, few studies consider mining events within a very long time span, e.g., ten years. In order to obtain a better global view of event evolution and understand the underlying causalities among events, it is necessary to analyze and mine events in a long time period.

To address these problems and challenges using mutli-type news streams with a long time period, this paper gives an overview of our Event Discovery and Retrieval System (EventSearch[1]) developed for facilitating users with better utilities to access news information. EventSearch has two major functions: 1) automatically detect events from multi-type historical data; 2) maintain a two-layer index and provides flexible and efficient retrieval operations to users. Compared with previous event search engines, our system has three significant merits:

- Multi-type data aggregation. We consider a broad aggregation of four type of news streams as our input and provide a very comprehensive perspective of news events.
- Very long news streams. Our system detects and traces events in a very long time period, i.e., ten years, and it provides a better understanding of event evaluation and causalities among events. e.g., Iraq war. Note although our dataset has covered a very long time span, our system is capable to extract event information accurate to days.
- Multi-modality event visualization. Our system provides a multimedia-based visualization for automatically discovered events. Each event snippet in our system consists of mutli-modality data elements, including text, photos and videos. These types of data are combined in a systematic way to enhance the readability of information and improve user utility.

2. SYSTEM ARCHITECTURE

We first introduce the system architecture of our demo EventSearch. It consists of a Back-End and a Front-End layer (Figure 1).

*Corresponding author

[1] Demo address: http://162.105.205.253/eventsearch/

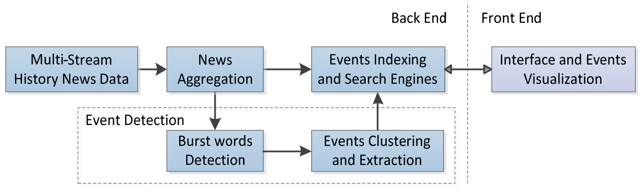
Back End
Front End
Multi-Stream History News Data
News Aggregation
Events Indexing and Search Engines
Interface and Events Visualization
Event Detection
Burst words Detection
Events Clustering and Extraction

Figure 1: The system architecture.

Back-End module mainly takes charge of data aggregation, data preprocessing, offline event extraction and index building. **1) Data aggregation.** We employ an aggregation engine to collect various news-related data. Our input contains four important sources of news-related data: Web page based, video based, traditional newspaper based and micro-blog based. The Web page based news articles come from InfoMall[2] system, which achieved billions of Chinese Web pages since 2001. Since our major focus is to mine newsworthy events, we extract 11M news articles from 2001 to 2011, by requiring that the URL field should contain the keyword "news". Video based news articles are mainly downloaded from CCTV[3], which is China's national daily news program. We download the videos and their corresponding textual information. For traditional newspapers, we partially downloaded the articles from RMRB[4], which is the official newspaper of the Communist Party of China. We totally extract 59K and 0.89M articles respectively for CCTV and RMRB. Those data also range from 2001 to 2011. For micro-blogs, we use an official API authorized by Sina Weibo[5], which is the largest micro-blog in China, and we can retrieve short micro-blog messages since 2009. We do not download any data from Sina Weibo but get the data by submitting real-time queries when needed. Specifically, we first perform event extraction from news articles and generate event keywords. Then we query Sina Weibo API with event keywords within the event period to generate micro-blog results when presenting the event. **2) Data preprocessing.** After collecting data, the aggregation engine assigns a unique global document ID to each document, and extracts different fields (URL, title, body, score, ID, timestamp, etc). Note that the definition of "document" is very general, not limited to textual articles. For example, one document may contain both videos and text. For each article, we extract its corresponding keywords and location information. For video based documents, we adopt a non-broadcast clips extraction algorithm[6] to extract representative pictures as descriptions of documents. Then we merge all the textual news data (including the text from video based documents) into one unified stream ordered by timestamps for downstream tasks. **3) Offline event extraction.** Our event algorithm consists of three major parts: a) detect bursty feature (A bursty feature comprises a bursty term together with a bursty period) from the unified stream; b) use these bursty features to represent documents for clustering; c) perform clustering based on our proposed text representation. See Section 3 for details. Note that we only perform event extraction on textual data, we map non-textual data to corresponding events by utilizing the relationship between textual data and non-textual data, i.e., they coexist in the same document.. **4) Index building.** When document are clustered into events, we further generate event-dependent features, including event keywords, location and start/end time. We design a two-layer index for the whole data collection, i.e., event-level index and document-level index. See Section 4 for details.

In the Front-End module, EventSearch provides a friendly Web based interface. To help users to flexibly explore the discovered events, we set up three kinds of query fields: keywords, time and location, and each kind of query field can be left as blanks. The search result consists of a set of event snippets. In addition to news titles, each event snippet contains rich event metadata, including event location, time, keyword clouds and photos. The search result interface also allows user to click the returned events to zoom into the event detail page. See Section 5 for details.

3. OFFLINE EVENT EXTRACTION

One standard way for event detection is to cluster news articles as events by following a two-step approach [5]: 1) represent document as vectors and calculate similarities between documents; 2) run the clustering algorithm and determine events. Although it seems very natural and straightforward, event detection in long text streams is very challenging. Traditional Vector Space Model does not consider modeling temporal aspects of documents and it is not suitable for temporal modeling. In addition, we have to face with the scalability problem from our very big news archive for clustering these temporal documents. To effectively and efficiently model temporal document for clustering, we propose a novel burst-based document representation.

Burst based VSM. Inspired by the work [2,3], we select *bursty features* as the basic representation units of documents, which contain both temporal and semantic information. We use *bursty feature* to denote a sudden surge of the frequency of a single term in a text stream, and represent a bursty feature by the term itself together with the time interval during which the burst takes place. For example, (Olympic, Aug-08-2008, Aug-24-2008) can be regarded as a bursty feature. We also call the term in a bursty feature its bursty term. We merged all the text documents as one news stream, and use the two-state machine algorithm [3] to detect a set of bursty features from this news stream, denoted as *B*. A document $\boldsymbol{d_i}(t)$ with timestamp *t* is represented as a vector of weights in ***bursty feature dimensions*:**

$$d_i(t) = (d_{i,1}(t), d_{i,2}(t), \ldots, d_{i,|B|}(t)) \qquad \textbf{(1)}$$

We define the *j*th weight of $\mathbf{d}_i$ as follows:

$$d_{i,j} = \begin{cases} \text{tf-idf}_{i,w_{B_j}}, & \text{if } t \in bi_{B_j} \\ 0, & \text{otherwise} \end{cases} \qquad \textbf{(2)}$$

where B_j is the *j*th bursty feature in *B*, w_{Bj} and b_{iBj} are the bursty term of B_j and bursty interval of B_j respectively, $|B|$ is the number of bursty features in *B*. When the timestamp of $\boldsymbol{d}_i$ is in the bursty interval of B_j and contains term w_{Bj}, we set up the weight using the common used *tf-idf* method.

From the definition above, we can see that one dimension is active only when the document falls in the corresponding bursty interval. BurstVSM considers both semantic and of temporal information: semantics are modeled by the weight of bursty terms while temporal dynamics are modeled by mapping one dimension to one bursty feature. Usually, BurstVSM represents one document as a vector with only a few non-zero entries. It makes computation of document similarities more efficient in a large dataset compared with VSM. See [7] for more details.

Event detection algorithm. Based on the discussions above, each document can be represented as a burst-based vector. We use cosine function to compute document similarities. Our dataset

[2] http://www.infomall.cn/

[3] http://cctv.cntv.cn/lm/xinwenlianbo/

[4] http://paper.people.com.cn/

[5] http://weibo.com/

consists of 11 million documents in a long time span (10 years), and it is infeasible to cluster all the documents together. We develop a heuristic clustering algorithm for event detection, denoted as *split-cluster-merge*. It includes three main steps, namely split, cluster and merge. The idea is that we first split the dataset into small parts, then cluster the documents of each part independently and finally merge similar clusters from two consecutive parts. It has a merit that *cluster* step could be executed in parallel for different parts. In our dataset, we find that most of events last no more than one month, so we split the dataset by months. After splitting, clustering can run in parallel for different parts (we use *CLUTO*[6] as the clustering tool), which significantly reduces total time cost. For *merge*, we empirically set a threshold of 0.5. The final clusters are returned as identified events. See [7] for more details.

4. EVENT INDEXING AND SEARCH ENGINE

For an event retrieval system, a user may submit either event-level queries (by inputting queries to retrieve relevant events) or document-level queries (by zooming into the detailed content of one specific document). To facilitate both types of queries, we build a two-layer index: document-level and event-level index.

Our textual data are indexed in the document-level index, including keywords, bursty features and corresponding event ID (in our system, one article only belongs to one event), timestamp and relevant locations. In our system, since documents may contain non-textual information, to manage them, we set up special fields in document-level index, e.g., photo linking field and video linking field. These fields are filled with internal position pointers on disks where we store non-textual information. To discriminate documents from different source types, we add a source type field in document-level index. In the event-level index, we store information from offline-discovered events, including start/end time, relevant locations and event keywords. One important field in even-level index is the collection of relevant documents in one event. Given a specific event, this field records all the global IDs of relevant documents. We use this field to link event-level index with document-level index. The two-layer index can support various queries flexibly.

Next, we describe how to perform keywords based retrieval using such a two-layer index. Figure 2 shows the general flow of query processing. Firstly, we use a BM25 like retrieval model for keyword-based queries in the document-level index. We keep top ranked 10000 articles and get their corresponding event IDs.

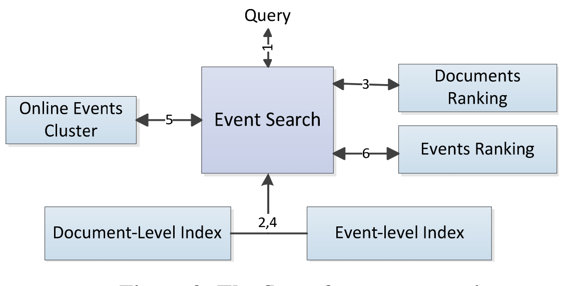

Figure 2: The flow of query processing.

Secondly, we obtain corresponding detailed information of these events from event-level index using retrieved event IDs, and these event details are used as ranking features, including the number of relevant documents (scales), event period (recency and time span), the number of different types and the number of related document in certain source. Thirdly, we compute the divergence of the distribution of relevant documents in different events. If the divergence is too large, in other words, relevant documents appear in quite a lot of offline-discovered events, we employ an online event extraction module to rediscover events from top ranked documents to better satisfy user needs. Otherwise we will combine relevant documents from the same event as an event result set. Finally, we use corresponding relevance scores or other criteria, e.g., recency, to rank events.

For non-keyword queries, we only use event-level index to retrieve events, and rank the candidate events by considering three factors: event static quality score (offline calculated), event period and event locations. In this case, online cluster module will not be triggered. Another important problem is how to incorporate the micro-blog data in the event snippet. Since we have generated event keywords and event period, for a given event, we query Sina Weibo API with the top ranked keywords within the corresponding period, and micro-blog results are generated in an online way.

5. EVENT VISUALIZATION

Our system allows the users to specify the queries by inputting three kinds of information: keywords, start/end time and location. When a user submits a query, the system will return a list of snippets for relevant events (shown in Figure 4). An event snippet consists of news titles, event keyword cloud, event period and relevant event locations, possibly with news photos and videos. Users can slip mouse on photos to select the corresponding videos, and our system will play the clicked video in a separate page (shown in figure 3).

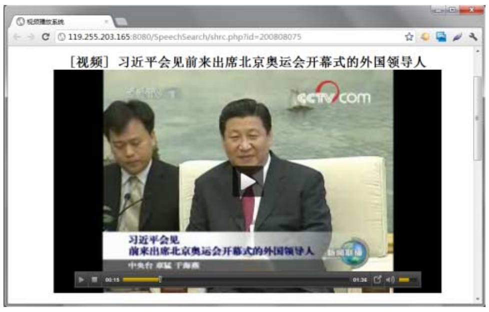
119.255.203.165:8080/SpeechSearch/shrc.php?id=200808075
[视频] 习近平会见前来出席北京奥运会开幕式的外国领导人
cctv.com
习近平会见
前来出席北京奥运会开幕式的外国领导人
00:15
01:36

Figure 3: Web page for playing video.

The major merit of our event visualization is that we incorporate multi-modality and multi-type information. Such visualizations can provide a comprehensive view of events and significantly improve user utilities. In addition, we have offered some basic mechanisms for mining and analyzing events. See the top two subfigures in Figure 4, respectively the trend of the number of events and the hottest relevant locations related with the given query. Take the query "Olympics Games" as one example (in Chinese), we can see that there is a significant spike in terms of the number of related events. The major reason is that the hold of "Beijing 2008 Summer Olympics Games". We use Baidu map to present the hottest event locations. In the subfigure on the right, Beijing has been marked as the hottest event location and Shanghai as second. More interestingly, users can click any points in the timeline or map to see the events that happened around that time or location.

Given these returned event snippets, a user may be interested in a specific event. Our system provides a link to event detail page to help users to zoom into the details of one event. Figure 5

[6] www-users.cs.umn.edu/~karypis/cluto/

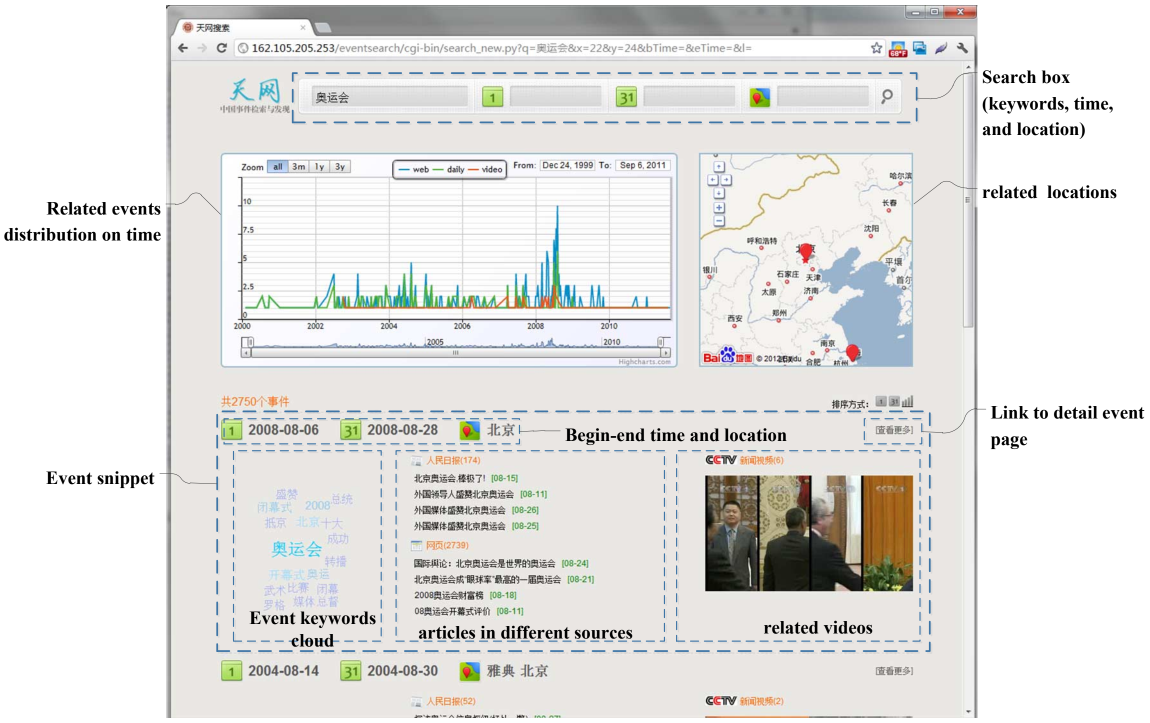
Search box (keywords, time, and location)
related locations
Related events distribution on time
Link to detail event page
Begin-end time and location
Event snippet
Event keywords cloud
articles in different sources
related videos
162.105.205.253/eventsearch/cgi-bin/search_new.py?q=奥运会&x=22&y=24&bTime=&eTime=&l=
奥运会
共2750个事件
2008-08-06
2008-08-28
北京
2004-08-14
2004-08-30
雅典 北京
人民日报(174)
网页(2739)
北京奥运会,棒极了! [08-15]
外国领导人盛赞北京奥运会 [08-11]
外国媒体盛赞北京奥运会 [08-26]
外国媒体盛赞北京奥运会 [08-25]
国际舆论:北京奥运会是世界的奥运会 [08-24]
北京奥运会成"眼球率"最高的一届奥运会 [08-21]
2008奥运会财富榜 [08-18]
08奥运会开幕式评价 [08-11]

Figure 4: Result page of event searching for query "Olympic Games".

presents an event detail page of “Beijing 2008 Summer Olympics Games”. Users can browse articles by selecting different views of by clicking source type tag i.e. in a specific type of news stream. Note that the keyword clouds shown on the left in Figure 5 are source dependent, indicating the system generates different keyword clouds for multiple sources. Relevant articles are returned and presented as document snippets, and we can rank those articles either by relevance or recency. Users can click the link in one document snippet to view the full information and corresponding link in InfoMall for one document (shown in figure 6).

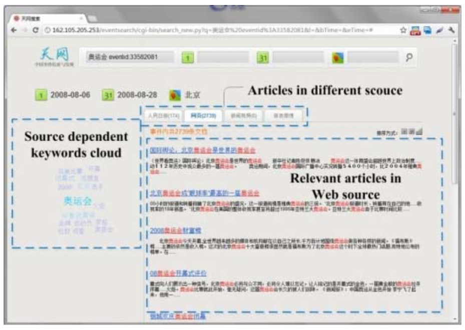
Articles in different scouce
Source dependent keywords cloud
Relevant articles in Web source

Figure 5: Event detail page for one event for query "Olympic Games".

Figure 6: News article's detail page and its archived page in InfoMall.

ACKNOWLEDEMENTS

This work is supported by NSFC Grant (60933004, 70903008, 61073082) and HGJ Grant 2011ZX01042-001-001.

6. REFERENCE

[1] Fung, G.P.C., Yu, J.X., Liu, H., and Yu, P.S. Time-dependent event hierarchy construction. *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM (2007), 300-309.

[2] He, Q., Chang, K., Lim, E.P., and Zhang, J. Bursty feature representation for clustering text streams. *Proc. SIAM Conference on Data Mining*, (2007), 26-28.

[3] Kleinberg, J. Bursty and hierarchical structure in streams. *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM (2002), 91-101.

[4] Mathioudakis, M. and Koudas, N. TwitterMonitor: trend detection over the twitter stream. *Proceedings of the 2010 international conference on Management of data*, ACM (2010), 1155-1158.

[5] Yang, Y., Pierce, T., and Carbonell, J. A study of retrospective and on-line event detection. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, ACM (1998), 28-36.

[6] Ye, Q., Huang, Q., Gao, W., and Jiang, S. Exciting event detection in broadcast soccer video with mid-level description and incremental learning. *Proceedings of the 13th annual ACM international conference on Multimedia*, ACM (2005), 455-458.

[7] Zhao, X., Chen, R., Fan, K., Yan, H., and Li, X. A Novel Burst-based Text Representation Model for Scalable Event Detection. In *ACL*, 2012.

.