

Event Detection from Evolution of Click-through Data*

Qiankun Zhao^b Tie-Yan Liu[‡]
^bPennsylvania State University
[†]Nanyang Technological University, Singapore
 {pg04327224,assourav}@ntu.edu.sg

Sourav S Bhowmick[†] Wei-Ying Ma[‡]
[‡]Microsoft Research Asia
 HaiDian District, Beijing, China 100080
 {tyliu,wyma}@microsoft.com

ABSTRACT

Previous efforts on event detection from the web have focused primarily on web content and structure data ignoring the rich collection of web log data. In this paper, we propose the first approach to detect events from the *click-through data*, which is the log data of web search engines. The intuition behind event detection from click-through data is that such data is often event-driven and each event can be represented as a set of *query-page pairs* that are not only *semantically* similar but also have similar *evolution pattern* over time. Given the click-through data, in our proposed approach, we first segment it into a sequence of bipartite graphs based on the user-defined *time granularity*. Next, the sequence of bipartite graphs is represented as a *vector-based graph*, which records the semantic and evolutionary relationships between queries and pages. After that, the vector-based graph is transformed into its *dual graph*, where each node is a query-page pair that will be used to represent real world events. Then, the problem of event detection is equivalent to the problem of clustering the *dual graph* of the vector-based graph. The clustering process is based on a two-phase *graph cut* algorithm. In the first phase, query-page pairs are clustered based on the *semantic-based similarity* such that each cluster in the result corresponds to a specific topic. In the second phase, query-page pairs related to the same topic are further clustered based on the *evolution pattern-based similarity* such that each cluster is expected to represent a specific event under the specific topic. Experiments with real click-through data collected from a commercial web search engine show that the proposed approach produces high quality results.

Categories and Subject Descriptors

H.3.3 [Information System]: Information Storage and Retrieval—*Relevance feedback*

*This work was done when the first author was doing internship at Microsoft Research Asia and was a PhD candidate at Nanyang Technological University, Singapore.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

KDD'06, August 20–23, 2006, Philadelphia, Pennsylvania, USA.
 Copyright 2006 ACM 1-59593-339-5/06/0008 ...\$5.00.

IP address	Query	Page	Time
xxx.xxx.xxx	BMW	http://www.bmw.com	14:02 02/11/2005
xxx.xxx.xxx	SVM	http://svm.first.gmd.de	15:31 03/25/2005
xxx.xxx.xxx	MSN	http://www.MSN.com	21:14 02/14/2005

Table 1: Example of the click-through data.

General Terms

Measurement, Algorithms, Experimentation

Keywords

click-through data, event detection, dynamic web, evolution pattern

1. INTRODUCTION

The web has invaded our lives. Web data now covers almost every object and event in the real world. That is, in some sense, the web is a sensor of the real world. This sensor-centric view of the web has recently triggered research efforts to extract knowledge such as topics, events, and stories from large volumes of web data [1, 9, 18, 19]. These approaches can be classified into two groups: *structure-based* extraction and *content-based* extraction. In the structure-based approaches, the website structures, hyperlink structures, and URLs are used to extract sets of web pages corresponding to events and objects [12, 8]. In the content-based extraction, content of web pages are segmented and categorized into subgroups that correspond to different topics, events, and stories using techniques such as natural language processing and probability models [18, 19]. At the same time, such extraction results have been proved useful in many applications such as organizing the website structure [12], restructuring the web search results [8], terrorism event detection [14], and *Photo Story* and *Chronicle* [9].

1.1 Motivation

In the context of event detection from web data, the *types* of data that are used play an important role. Web data can be broadly classified into two types: *author-centric* and *visitor-centric*. *Author-centric* data refers to a set of hyperlinked web pages that describes certain object or event, while *visitor-centric* data refers to the web log data. Author-centric data is created by web publishers for user browsing and represents web content and structure data. On the other hand, the visitor-centric data is generated as a result of users' browsing activities or query activities (for search engines). Observe that author-centric data describes authors' point of view while visitor-centric data reflects the web visitors' point of view.

We observed that in most of the existing event detection approaches only the author-centric data is considered while

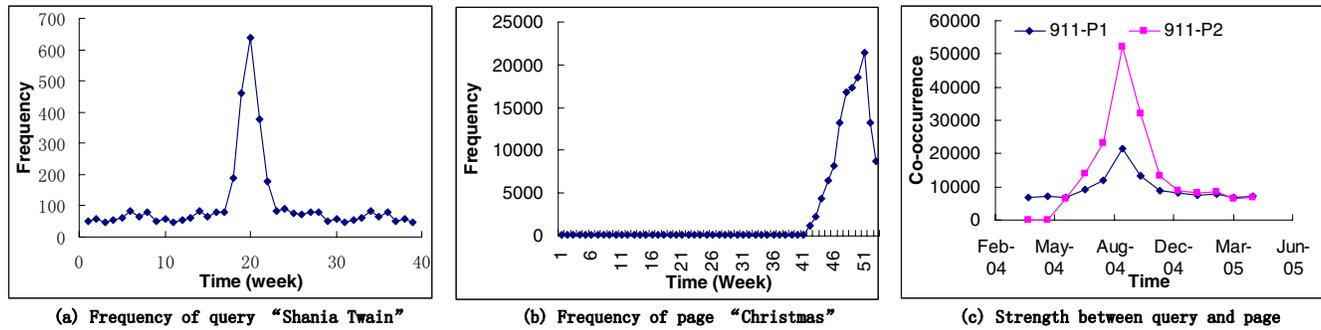


Figure 1: Examples of click-through data dynamics.

the rich collection of visitor-centric data is ignored. We believe that in many real life cases, the visitor-centric data plays equally important role for event detection. *In this paper, we propose to detect events from the visitor-centric data.* Note that visitor-centric data can be either web access patterns of visitors reflecting their browsing activities or it can be *click-through data*, which is the web search engine log data [6] that records the interactions between users and a web search engine. Table 1 shows the standard format of the click-through data extracted from a commercial web search engine. Observe that the click-through data usually contains the IP address of the user who issued the query, the query terms, the web pages that were clicked in the search results, and the corresponding timestamp of the query. Note that due to privacy issue, we do not show the exact IP address in this table. In this paper, we focus our attention to detect events from the click-through data. *To the best of our knowledge, this is the first effort to detect events from click-through data.* We show that it is indeed possible to produce high quality event detection results *without* analyzing the content and structure of web pages (author-centric data). Note that, in [17], events are defined as sets of Web pages. In contrast, as we are detecting events from rich click-through data, we can exploit the query keywords as well as related pages that are clicked as representation of events. Note that the query keywords can be used as “summary” of the corresponding events.

As previous works on event detection from web data have primarily focused on detecting events from a collection of hyperlinked pages, we first justify the reasons for choosing the web search engine click-through data as the source for event detection. First, the increasing popularity of web search engines has given rise to a large number of search engine users issuing huge volumes of queries. These queries often return links to high quality web pages. Consequently, there is a large volume of click-through data that can be potentially exploited for event detection. Second, as shown in Table 1, the click-through data contains the query keywords that are created by users and links to web pages that often describe real world events. Specifically, as we shall see later, these keywords and the corresponding pages clicked by the users often reflect their response to recent real world events.

Besides exploiting visitor-centric data, another key distinguishing feature of our proposed event detection approach is that *we take into account the dynamic nature of the visitor-centric data in the event detection process.* Here, *dynamic* nature refers to the evolving nature of the queries and pages in the click-through data over time. For

instance, visitors may formulate new queries that were not formulated before, or some previously executed queries may be executed more or less frequently due to some recent events. Furthermore, new web pages that were not available earlier may now appear in the search results and hence they may be clicked now by visitors. Similarly, previously clicked pages may be clicked more or less frequently in recent times. Also, previously executed queries may lead to a different set of web pages when the queries are formulated during a specific time period. As a result, the frequencies of queries being issued and pages being clicked may change over time. Let us illustrate this with examples. Figure 1(a) shows how the frequency of a query keyword “Shania Twain”¹ changes over time, where the y -axis represents the frequency of the query being issued during the corresponding week in the x -axis. Similarly, Figure 1(b) shows how the frequency of the web page at *www.christmas.com* being accessed via a specific search engine changes over time. More importantly, the *query-page relationship* may change over time. Hereafter, *query-page relationship* refers to the *co-occurrence* of the query and page pair, which is the number of times a specific query and a page appear in the same row in Table 1. A large co-occurrence value indicates that the query-page pair occurs more frequently in the click-through data. Figure 1(c) shows the query-page relationships of two *query-page pairs* “911- P_1 ” and “911- P_2 ” over time where P_1 is the homepage of *www.911digitalarchive.org* and P_2 is the homepage of *www.fahrenheit911.com*.

Our investigation in [20] with real click-through data revealed that the above dynamic behavior of the click-through data is often driven by real world events. Specifically, the frequencies of the issued queries and clicked pages as well as the co-occurrences of the queries and pages often depend on real world events that have happened, are happening, or are going to happen. For example, reconsider Figure 1(c). In September 2004, the frequency of the query “911” and corresponding web page P_1 being issued and accessed increased dramatically. At the same time, the co-occurrence of “911” and P_2 increased substantially. Further investigation revealed that this is primarily due to the occurrences of two events during that time. One is the three year anniversary for the “911” event in 2001 (represented by P_1). The other is the new movie “Fahrenheit 911” (represented by P_2) which was released in June 2004 in the United States. Observe that co-occurrence of “911- P_2 ” started increasing from end of June and reached a maximum value in 09/2004.

¹The graphs in Figure 1 are based on real click-through data collected from a commercial search engine.

By incorporating such dynamic nature of the click-through data in the event detection process, the quality of events detected can be improved. Let us illustrate this with the following examples.

- When a new event occurs, the number of related queries being issued and the number of related web pages being visited may increase dramatically. At the same time, the co-occurrence relationships between these queries and pages are surprisingly strong. For example, we observed from the click-through data of a commercial search engine that the query of “Shania Twain”, a singer, was surprisingly popular from November 2004 to December 2004 as shown in Figure 1(a). Also, there is a set of related web pages that have high co-occurrences with this keyword. It turns out that “Shania Twain” was expected to have a series of concerts in the first two weeks of December, 2004 in the US. Observe that this event can be detected by analyzing the dynamic behavior of the frequency of the issued query, clicked web pages, and the query-page relationships.
- For a group of similar queries, the corresponding web pages being clicked can be substantially different from time to time. In another word, the same query may represent different events depending on the set of web pages clicked by the users in the query result set. Similarly, the same page may represent different events depending on the query that directed to this page. Let us reconsider the example in Figure 1(c). The query of “911” was issued frequently in September 2004 and a lot of pages about the 911 anniversary event and the new movie *Fahrenheit 911* were clicked. Using existing event detection approaches, which take the click-through data as snapshot data, it is difficult to distinguish the two events as they share a list of similar queries and the web pages being clicked may be connected via hyperlinks. However, by incorporating the dynamics of the query-page relationship over time, as shown in Figure 1(c), the two events can be easily distinguished.

1.2 Overview

Our proposed event detection approach works as follows. Given the click-through data, firstly, it is segmented into a sequence of collections based on the user-defined *time granularity*, where each collection is represented as a *bipartite graph*. Then, the sequence of bipartite graphs is merged to form the *vector-based graph*, which records the evolution of the relationships between queries and pages over time using vectors. After that, the vector-based graph is transformed into its *dual graph* such that each node is a query-page pair that will be used to represent the real world events. Then, the event detection problem is defined as the problem of clustering the dual graph into subgraphs such that the query-page pairs within each subgraph represent a real world event. We propose a two-phase *graph cut* algorithm that partition the dual graph based on the *semantic-based similarity* and *evolution pattern-based similarity* in turn. Details of the event detection framework will be discussed in Section 2. Experimental results with real dataset from a commercial web search engine show that the visitor-centric data can produce event detection results with high quality and vari-

ous types of events can be detected using the proposed approach. In summary, the main contributions of this paper are as follows.

- To the best of our knowledge, this is the first effort to detect events from visitor-centric and historical click-through data without taking into account the content and structure of individual web pages.
- We present a new graph model, called *vector-based graph*, to represent both the *evolution patterns* of the relationship between queries and pages and their *semantic* relationship in the click-through data. Furthermore, by transforming the vector-based graph to its *dual graph*, we present a novel representation of real world event using the subgraph of query-page pairs.
- A two-phase *graph cut* algorithm is proposed to cluster the dual graph representation of the click-through data into subgraphs that correspond to real world events using the semantic and evolution pattern-based *similarity measures*.
- Performance of the proposed event detection approach is evaluated with extensive experiments using a very large collection of real click-through data collected from a commercial web search engine.

The rest of this paper is organized as follows. In Section 2, we present the event detection framework. Then, Section 3 describes the representation of the click-through data. Similarity measures and the event detection algorithm are discussed in Sections 4 and 5, respectively. In Section 6 we study the performance of our proposed framework. Applications of the event detection results are presented in Section 7 and related works are reviewed in Section 8. The last section concludes this paper.

2. EVENT DETECTION FRAMEWORK

The framework of the event detection process, which consists of four major steps, is shown in Figure 2. We elaborate on each step in turn.

Given the click-through data and the user-specified time granularity such as hour, day, week, month, etc., the data is segmented into a sequence of collections such that the user sessions within each collection are within the same time interval. Note that the time granularity is user-defined and application-dependent. For example, if the click-through data is segmented on an hourly basis, then the click-through data for a particular day will be partitioned into a sequence of 24 collections. Then, each collection is represented as a bipartite graph $G = (V, E)$, where nodes in V represent queries and web pages and edges in E represent the *strengths* of the query-page pairs within the time interval [16] (defined in Section 3). As a result, the input click-through data is transformed into a sequence of bipartite graphs.

To represent the evolution of the query-page relationships, we propose to merge the sequence of bipartite graphs into a *vector-based graph*, where the edge in the graph is a vector that represents the sequence of strengths of the query-page relationship over time as shown in Figure 2 and the vertices represent the queries and web pages. For clarity, we only show the strength vector of one query-page pair in the figure. Note that the strength vector in the vector-based graph enables us to model the evolution pattern of the strength of query and page relationship. However, we cannot directly detect events from the vector-based graph by partitioning it into subgraphs for the following two reasons.

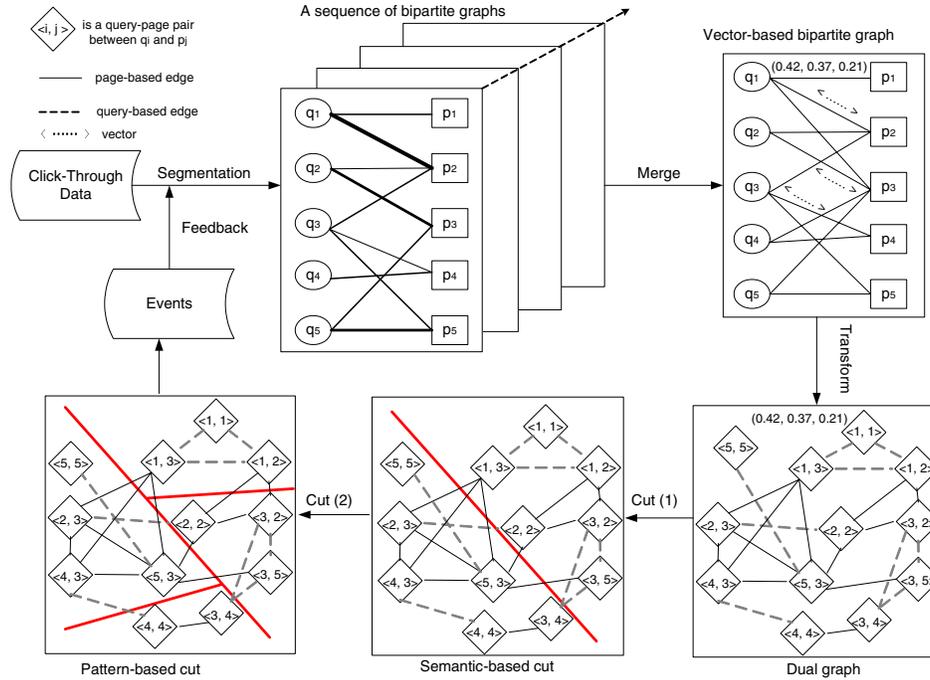


Figure 2: The Framework for event detection from click-through data.

- First, real world events are expected to be represented as sets of query-page pairs, while the vertices in the vector-based graph represent individual queries or web pages. More specifically, to detect events, we need to know the similarity between query-page pairs rather than the similarity between queries and pages.
- Second, some queries or pages may belong to two or more different events, as we discussed in the previous section (Figure 1(c)). If we partition the vector-based graph, then the query term(s) can *only belong to one* of the events as it is represented as a single vertex. Hence, queries or pages that represent multiple events pose a problem if we attempt to detect events directly from the vector-based graph.

Consequently, we transform the vector-based graph into its *dual graph*, which is well-defined in graph theory [4]. We shall elaborate on the transformation in Section 3. The basic idea is that the edges in the vector-based graph are transformed into vertices and the vertices in the vector-based graph are transformed into edges in the dual graph. Note that the vectors in the vector-based graph are represented as attributes of the vertices in the dual graph as shown in Figure 2. Note that for clarity, we only show the attribute of vertex $\langle 1, 1 \rangle$ in Figure 2. The dual graph addresses the above two issues in the following ways. In the dual graph, each vertex is a *query-page pair* and each edge represents the *similarity* between the query-page pairs. Moreover, the same query can be included in more than one vertex in the dual graph.

Then, the event detection problem can now be formulated as the problem of partitioning the dual graph based on the similarity values between query-page pairs. In this paper, we consider two types of similarities, the *semantic-based similarity* and the *evolution pattern-based similarity*. The *evolution pattern-based similarity* values are computed from the co-occurrence vectors while the *semantic-based similarity* values are computed from the structure of the dual graph.

We shall elaborate on these two metrics in Section 4. Based on the state-of-the-art graph cut algorithm [10] and the two similarity measures, a two-phase graph cut algorithm is proposed to partition the dual graph. Firstly, the dual graph is cut into subgraphs that correspond to groups of semantically related events using the semantic-based similarity. After that, the groups of semantically related events are partitioned again into individual events using the evolution pattern-based similarity. By looking into the event detection results, the users can further refine the time granularity with respect to the application requirements to detect different types of events.

In the subsequent sections, we shall focus on the following three issues: the vector-based graph and its dual graph representation of the click-through data, the semantic-based and evolution pattern-based similarity measures, and the event detection algorithm that cuts the dual graph using the proposed similarity measures.

3. DATA REPRESENTATION

In the literature, there are two different ways to represent the click-through data. One way is to represent the click-through data as a session database where each session represents a pair of a query and a page that the user clicked [15, 3] as shown in Table 1. Recently, some variants of this representation have been proposed by taking into account the rank of the page and the ID of the corresponding users [13, 6]. Another way of representing the click-through data is to represent it as a bipartite graph, where the queries and pages are nodes and the query-page pair relationships are the edges as shown in Figure 2 [2, 16].

However, we observed that previous representations of the click-through data, to the best of our knowledge, do not take the timestamps of the query-page pairs into account. That is, occurrences of a query-page pair are taken as equally important and meaningful even though the corresponding timestamps are different. This assumption may not always

be true in real life applications. For instance, reconsider the examples in Section 1. The co-occurrences of a query-page pair at different timestamps may be taken as different indicators of the underlying events. More importantly, the evolution patterns of the query-page relationships may indicate the evolution of the corresponding events. In this paper, we propose to monitor their evolutionary patterns and use such patterns for accurate event detection as well.

To represent the evolutionary patterns, the occurrences of the query-page pairs in the click-through data are partitioned into collections based on their timestamps and the user-defined time granularity as discussed in Section 2. Note that the time intervals can be defined according to users' requirement and the temporal properties of the interesting events. By doing this, the click-through data can be represented as a sequence of collections, where each sub-group can be represented as a bipartite graph, which is commonly used in click-through data analysis [16].

Formally, given a click-through data C , it is partitioned into a sequence of collections based on user-defined time granularity, denoted as $\langle c_1, c_2, \dots, c_n \rangle$. We define the *strength* of a query-page pair $\mathcal{P}_{s,t} = (q_s, p_t)$ in c_i , denoted as $s_i(\mathcal{P}_{s,t})$, as $s_i(\mathcal{P}_{s,t}) = \frac{|\mathcal{P}_{s,t}(c_i)|}{\sum_{i=1}^n |\mathcal{P}_{s,t}(c_i)|}$ where $1 \leq i \leq n$. Note that the numerator is the number of co-occurrences of the query-page pair in the collection c_i and the denominator is the total number of co-occurrences of the query-page pair in C . Observe that the strength measure is the ratio of the co-occurrences of a query-page pair in a specific collection c_i against the total number of co-occurrences of the query-page pair in the entire collection C .

If we consider the absence of a query and a page as a relationship of strength 0, then we can assume that the sizes of the sequence of bipartite graphs are equal and fixed, and the only difference between these graphs are the weights of the edges. Thus, the representation of the click-through data can be formulated as a merged bipartite graph, called *vector-based graph*. Formally,

DEFINITION 1. Let $\langle c_1, c_2, \dots, c_n \rangle$ be the sequences of click-through data segmented from click-through data C using the user-defined time granularity. A vector-based graph is a bipartite graph, $G = (V, E)$, where V is a set of queries and pages; E is a set of labelled edge (q, \mathbf{l}, p) where $q, p \in V$ and \mathbf{l} is a vector denoted as $[s_1, s_2, \dots, s_n]$ that represents the sequence of strengths of the query-page pair (q, p) where s_i is the strength of (q, p) in c_i . \square

For example, suppose the vector corresponding to the query-page pair (q_1, p_1) is $[0.42, 0.37, 0.21]$. It means that the *strengths* of the query-page pair (q_1, p_1) are $0.42, 0.37$, and 0.21 , respectively in the three consecutive time intervals. Note that different from previous approaches, the query-page relationships are decomposed into a vector. Each individual value represents the corresponding strength of the query-page relationship during a specific time period. As a result, we successfully embed the temporal and evolutionary information of the click-through data in the vector-based graph representation.

As discussed in the framework, we have to transform the vector-based graph to its *dual graph*. Basically, in the dual graph, each vertex is a query-page pair, which is an edge in the bipartite graph; and each edge is a query/page that links the two query-page pairs, which is a vertex in the bipartite graph. Formally, dual graph is defined as follows.

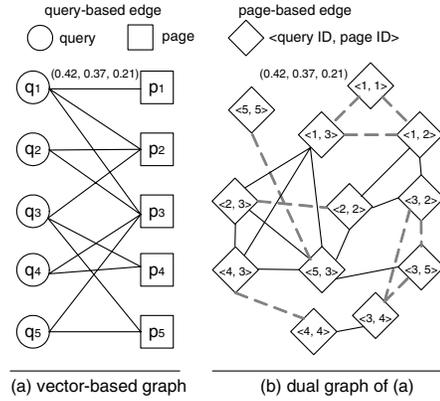


Figure 3: Example of vector-based and dual graphs.

DEFINITION 2. Given a vector-based graph $G = (V, E)$, a graph $G' = (V', E')$ is the dual graph of G if and only if 1) V' is a set of vertex labelled as (p, \mathbf{l}, q) and for each vertex $v'_i \in V'$ there is a corresponding edge in E ; 2) E' is a set of edges and for each $e'_{ij} \in E'$ there is a corresponding vertex in V ; 3) There exists an edge e_{ij} that connects v'_i and v'_j if and only if $q'_i = q'_j$ or $p'_i = p'_j$. \square

An example of the vector-based bipartite graph and the corresponding dual graph is shown in Figure 3. Note that the dotted edges in the dual graph represent the *query-based edge*, while the solid edges represent the *page-based edge* in the vector-based graph. Here, a *query-based edge* connects two query-page pairs that share the same query, while a *page-based edge* connects two query-page pairs that share the same page. Moreover, in the dual graph, the vectors that represent the historical strengths of the query-page pairs are now represented as attributes of the nodes in the dual graph.

4. SIMILARITY MEASURES

As mentioned in the preceding section, there are two types of similarities concerning the query-page pairs. In this section, we elaborate on this. The intuition behind *semantic-based similarity* is that the query-page pairs sharing the query or page are similar to each other. One of the key features of the similarity is that it can be *propagated*. For example, suppose we have two query-page pairs $\mathcal{P}_{i,j} = (q_i, p_j)$ and $\mathcal{P}_{s,t} = (q_s, p_t)$ that share neither the query nor the page. However, $\mathcal{P}_{i,j}$ and $\mathcal{P}_{s,t}$ can be similar if there exist another query-page pair $\mathcal{P}_{i,t} = (q_i, p_t)$, which shares the query with $\mathcal{P}_{i,j}$ and the page with $\mathcal{P}_{s,t}$. The second similarity is the *evolution pattern-based similarity*. That is, query-page pairs sharing similar evolution patterns are expected to be similar. Specifically, evolution pattern refers to the pattern how the strength of the query-page pair relationships change over time in the history as shown in Figure 1(c). Typical evolution patterns include periodic change, burst, increasing or decreasing change, and other model-based changes. To represent the different types of similarities, we decompose the dual graph into two different graphs: *semantic-based graph* and *evolution pattern-based graph*, and then define the two types of similarities, respectively.

4.1 Semantic-based Graph and Similarity

The semantic-based graph is structurally the same graph as the dual graph except that the vector attributes in the

original dual graph are not included in the semantic-based graph. Each edge e_{ij} , if exists, in the semantic-based graph denotes the strength of the relationship between two query-page pairs that are connected by e_{ij} . Recall that in the dual graph there are two types of edges the *query-based edge* and the *page-based edge*. The semantic relationship between nodes is actually the fusion of this page-based and query-based relationship. For simplicity, we represent such a fusion as a linear combination of the two types of relationships, where the weights of the page-based and query-based edge are denoted as α and β respectively ($\alpha + \beta = 1$). In this paper, we set both α and β to 0.5 . Then, the semantic-based similarity between two nodes a and b in the dual graph, denoted as $S^s(a, b)$, is defined as follows.

DEFINITION 3. *Let a and b be two nodes in the dual graph. The semantic-based similarity between them, denoted as $S^s(a, b)$, is defined as:*

$$S^s(a, b) = \frac{C}{|N(a)||N(b)|} \sum_{i=1}^{|N(a)|} \sum_{j=1}^{|N(b)|} S^s(N_i(a), N_j(b))$$

where C is the constant decay factor between 0 and 1, $|N(a)|$ and $|N(b)|$ are the numbers of neighbors for node a and b , respectively. $N_i(a)$ and $N_i(b)$ are the i^{th} neighbors of $N(a)$ and $N(b)$, respectively. \square

From the above definition, it is obvious that the similarity value has a range between 0 and 1. In this paper, the decay factor is set to 0.7 . Also, the equation is recursive and the similarity between query-page pairs can be propagated at the next recursion. It has been proved that the value of S_k^s , which is based on the value of S_{k-1}^s , is non-decreasing and will converge eventually [5]. In our approach, we start with S_0^s :

$$S_0^s = \begin{cases} \alpha & \text{if } a \text{ and } b \text{ are connected via a query edge} \\ \beta & \text{if } a \text{ and } b \text{ are connected via a page edge} \\ 0 & \text{otherwise} \end{cases}$$

4.2 Evolution based Graph and Similarity

Similar to the semantic-based graph, in the evolution pattern-based graph, each vertex also represents a query-page pair. However, the edges in the evolution pattern-based graph are different. Specifically, an edge between two query-page pairs represents the evolution pattern-based similarity calculated based on the affiliated vectors. As a result, the evolution pattern-based graph is a fully connected graph. Since the dimension of the vectors can be huge, we propose to summarize the vector using histogram with different granularity such that the dimensions can be reduced. Then, similarity is defined based on the histogram representation of the vectors using dynamic time warping [7].

Suppose the click-through data is segmented into a sequence of n collections based on the user-defined time granularity. Then there will be a sequence of n strength values represented as a vector $\mathbf{S} = [s_1, s_2, \dots, s_n]$. Given the window size of the histogram representation, w , each vector will then be compressed into histogram representation $\mathcal{H} = (H_1, H_2, \dots, H_{n/w})$, where $H_i = \sum_{j=(i-1) \times n/w}^{i \times n/w} s_j$. Based on different window size, the histogram representation may show different characteristics of the data in different granularity. In this paper, we will show that different types of events can be detected using different width in Section 6.

Given two query-page pairs $\langle q_i, p_j \rangle$ and $\langle q_s, p_t \rangle$, denoted as $\mathcal{P}_{i,j}$ and $\mathcal{P}_{s,t}$, the histogram representations of their histor-

ical strengths are denoted as $\mathcal{H}^{i,j}$ and $\mathcal{H}^{s,t}$. In the dynamic time warping approach, an $n \times n$ matrix is built, where the $(x^{\text{th}}, y^{\text{th}})$ element denotes the distance between $H_x^{i,j}$ and $H_y^{s,t}$ (typically the Euclidean distance is used). A warping path, \mathcal{W} , is a set of continuous elements in the matrix that defines a mapping between $\mathcal{H}^{i,j}$ and $\mathcal{H}^{s,t}$. The k^{th} element of \mathcal{W} is defined as $w_k = (x, y)_k$, so we have $\mathcal{W} = \langle w_1, w_2, \dots, w_k \rangle$, where $n \leq k \leq 2n - 1$. The warping path is typically subjected to the following three constraints.

- **Boundary conditions:** the warping path should starts from $(1, 1)$ and ends at (n, n) .
- **Continuity:** given $w_k = (a, b)$ and $w_{k-1} = (a', b')$, then $a - a' \leq 1$ and $b - b' \leq 1$.
- **Monotonicity:** given $w_k = (a, b)$ and $w_{k-1} = (a', b')$, then $a - a' \geq 0$ and $b - b' \geq 0$.

Then, formally the evolution pattern-based similarity is defined as follows:

DEFINITION 4. *Let $\mathcal{P}_{i,j}$ and $\mathcal{P}_{s,t}$ be two query-page pairs with dynamic warping $\mathcal{W} = \langle w_1, w_2, \dots, w_k \rangle$. The evolution pattern-based similarity between $\mathcal{P}_{i,j}$ and $\mathcal{P}_{s,t}$, denoted as $S^P(\mathcal{P}_{i,j}, \mathcal{P}_{s,t})$, is defined as:*

$$S^P(\mathcal{P}_{i,j}, \mathcal{P}_{s,t}) = 1 - \min \left\{ \frac{1}{k} \sqrt{\sum_{m=1}^k w_m} \right\} \quad \square$$

It can be observed that if two query-page pairs have exactly the same evolution pattern, then the evolution pattern-based similarity will be 1. From the definition, we can observe that the range of the similarity is between 0 and 1.

5. EVENT DETECTION ALGORITHM

In this section, we present the algorithm for event detection from the dual graph representation of the click-through data. We first briefly introduce the normalized graph cut algorithm [10]. Then, a two-phase clustering algorithm is proposed to cluster the dual graph based on the semantic-based similarity and evolution pattern-based similarity using the normalized graph cut algorithm.

5.1 Normalized Graph Cut

Given a graph $G = (V, E)$, suppose the nodes of the graph are points in the feature space, and an edge is formed between every pair of vertices. The weight of each edge, $w(i, j)$ represents the similarity between vertex i and vertex j . In graph theory, graph cut problem is to partition the set of vertexes into disjointed sets V_1, V_2, \dots, V_m based on certain similarity metric such that the similarity among vertices within V_i is high and across different sets V_i, V_j is low. As presented in [11], the problem of graph cut is decomposed into a sequence of recursive bi-partition processes and there are different measures for the bi-partition process. In this paper, we use the normalized graph cut [10], which has been widely used in object detection from video and image data.

A graph $G = (V, E)$ can be partitioned into two disjointed sets A and B by removing edges connect the two sets such that $A \cup B = V$ and $A \cap B = \emptyset$. The similarity between the two sets can be computed as the total weights of the edges that have been removed. In graph theoretic language, it is called *cut*:

$$cut(A, B) = \sum_{u \in A, t \in B} w(u, t)$$

To avoid the unnatural bias for partitioning out small sets of points, the normalized graph cut was proposed [10]. Instead of looking at the value of total edge weight connecting the two partitions, normalized graph cut computes the cut cost as a fraction of the total edge connections to all the nodes in the graph. The disassociation measure of *normalized cut* is denoted as $Ncut$:

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)}$$

where $assoc(A, V) = \sum_{u \in A, t \in V} w(u, t)$ is the total connections from vertex A to all vertices in the graph and $assoc(B, V)$ is similarly defined. Thus, the problem is to minimize the $Ncut$ value, which is NP-complete and solved using the eigenvectors for the following equation [10]:

$$(D - W)x = \lambda Dx$$

where x is $N = |V|$ dimensional vector, $x_i=1$ if vertex i is in A , otherwise $x_i=-1$. D is an $N \times N$ diagonal matrix with d_i on its diagonal where $d(i) = \sum_j w(i, j)$ is the total connection from vertex i to all other vertexes. The variable W in the above equation is an $N \times N$ symmetrical matrix with $W(i, j) = w_{ij}$. It has been proved that using the eigenvector with the second smallest eigenvalue can provide good partition results. As there are more than two partitions in our problem, we adopt the recursive 2-way cut algorithm, which works as follows.

5.2 Event Detection Algorithm

The algorithm of event detection from the click-through data is shown in Algorithm 1. Basically, as highlighted in the framework in Figure 2, the event detection algorithm consists of four major steps. Given the click-through data, the first step is to segment the query-page sessions into collections based on the user-defined time granularity. Each collection is represented as a bipartite graph and the click-through data is represented as a sequence of bipartite graph as shown in Lines 1-3 in the algorithm. Then, the sequence of bipartite graphs is merged into a single graph called vector-based graph. As a result, all the evolution patterns of the query-page relationships are stored. After that, the vector-based graph is transformed into its dual graph, where each vertex is a query-page pair. The goal is then to partition the dual graph into subgraphs that represent real world events. The partition algorithm is based on the above normalized graph cut as shown in Lines 5-10 in the algorithm. The graph cut algorithm is applied to the dual graph twice. Firstly, the graph is cut into subgraphs based on the semantic-based similarity such that each subgraph may contain a set of semantically related events. Then, the subgraphs are cut again based on the evolution pattern-based similarity.

Note that here we focused on designing algorithm that can detect sub-groups of click-through data corresponding to real world events. As we shall see in Section 6.3, there are different types of events such as periodic events (national holidays), burst events (unpredictable accidents), etc. The issue of *classifying* the detected events further into different categories will be part of our future work.

6. PERFORMANCE EVALUATION

In this section, we study the performance of our event detection approach. Firstly, the characteristics of the dataset

Algorithm 1: Event Detection from Click-Through Data

Input: A set of query-page sessions: D

Output: A set of query-page clusters: C_1, C_2, \dots, C_n

- 1: Partition the query-page sessions into a sequence of groups $\langle G_1, G_2, \dots, G_n \rangle$ based on the user-defined time interval
 - 2: Construct the corresponding bipartite graphs $\langle B_1, B_2, \dots, B_n \rangle$
 - 3: Construct the vector-based bipartite graph B_v
 - 4: Construct the dual graph D_B from B_v
 - 5: Calculate the semantic-based similarity among vertexes in D_B
 - 6: Perform the semantic-based normalized graph cut on D_B
 - 7: Store the intermediate clusters $C^i = \{C_1, C_2, \dots, C_m\}$
 - 8: Calculate the pattern-based similarity for vertexes within the same cluster in C^i
 - 9: Perform the pattern-based normalized graph cut to individual cluster in C^i
 - 10: **Return the updated clustering results C_1, C_2, \dots, C_n**
-

are presented. Then, quality of the experimental results is evaluated under different scenarios. Moreover, a list of events detected from the dataset is presented.

6.1 Dataset

A real click-through dataset that is extracted from MSN search engine is used in the following experiments. The click-through data contains 15 billion records of query-page pairs over 32 days from June 16, 2005 to July 17, 2005. The entire click-through data is partitioned into 768 collections, where each collection consists of the query-page pairs occurred during an interval of one hour. The average number of query-page pairs in each group is around 2,100,000.

6.2 Quality of Detected Events

To evaluate the quality of the event detection results, we manually labelled a list of 30 representative real world events as test cases. The reason is that for event detection from click-through data there is no benchmark dataset. In our experiment, each labelled event consists of a set of query-page pairs. The labelling process is as follows.

1. Select a list of *key queries*, denoted as \mathcal{K}_q , which was frequently issued and corresponded to a list of specific events.
2. For each key query $q_i \in \mathcal{K}_q$, a set of *key pages*, denoted as \mathcal{K}_p , that is led to by q_i is extracted.
3. For each key page $p_i \in \mathcal{K}_p$, the list of queries that led to p_i is extracted and inserted into \mathcal{K}_q .
4. Recursively execute step 2 and 3 to update \mathcal{K}_p and \mathcal{K}_q till the frequencies of the queries and pages extracted reach the minimum threshold.
5. Based on the frequency of the queries and pages, together with their relationships in the click-through data, we manually select a list of query-page pairs to represent each event referring to the related news archives that record real world event.

Some of the selected events and corresponding queries are shown in Table 2. Here a *key query* is a query that has been issued most frequently and lead to most of the event relevant pages. The *expanded queries* are these queries that were issued frequently and lead to relevant pages in the labelled events. The timestamp represents the time interval when the queries were issued. For example, the key query “*fireworks*” is the most frequently issued query relevant to the firework show during the Independence Day of the United States. From the expanded queries, it can be observed that people are preparing for that event. For another example, the key query “*Lohan*” is about an entertainment star in the

Key query	Expanded queries	Timestamp
fireworks	July-4th show picture foto locations buy cheap black market safety injuries	June 21, 2005- June 27, 2005
Wimbledon	tennis championships results women's 2005 live schedule	June 21, 2005- June 27, 2005
Lohan	Lindsey boob jay-leno tonight-show picture	June 21, 2005- June 27, 2005
battlefield	2 player joystick cracked server code rent keygen download review demo tips	June 21, 2005- June 27, 2005
earthquake	California center current recent picture locations	June 13, 2005- June 20, 2005
batman begins	Wallpaper review picture download scarecrow cast show movie walkthrough	June 13, 2005- June 20, 2005
Live 8	concert song pink pictures floyd download locations list	June 27, 2005- July 04, 2005
War of the world	movie photo rating review game download cruise wells	June 27, 2005- July 04, 2005
hurricane dennis	forecast update path 2005 weather projected report track	July 04, 2005- July 11, 2005
kelly monaco	naked playboy-news movie pics nude gallery biography	July 04, 2005- July 11, 2005
british open	odds golf history score tiger results leader coverage 2005	July 11, 2005- July 18, 2005
harry potter	half-blood-prince party shop book song goblet spoilers release summary	July 11, 2005- July 18, 2005
Wedding Crashers	movie quotes review script premier trailer diora	July 11, 2005- July 18, 2005

Table 2: Examples of events.

	in event	not in event
in cluster	a	b
not in cluster	c	d

Table 3: The cluster-event contingency table.

US, during the week she attended the “jay leno show”. As a result, many related queries were issued most likely by her fans. It can be observed that there are two types of scenarios. First, the queries may be issued frequently with many visits to the pages *before* scheduled events such as Independence Day of the US. Second, the queries are issued *after* certain unpredictable events such as the recent earthquakes and hurricanes.

Note that the experiment is conducted with the entire click-through data, which may produce many other events that are not listed in the table. The reason that we do not select only the query-page pairs related to the selected event is that, in most case, these events are not connected in the bipartite representation of the click-through data.

With the partitioning results of the click-through data, the clusters that *best match* the 30 labeled events are used for evaluation. Note that both the clusters in the event detection results and the labelled events are represented as sets of query-page pairs. A *best match* of a labelled event is a cluster of query-page pairs in the event detection results that share the same *key query* and has the maximum overlap between the labelled event in terms of the number of common query-page pairs. Then, the evaluation is based on the *contingency tables*.

Table 3 shows the two-by-two contingency table for a cluster-event pair, where a , b , c , and d are the numbers of query-page pairs in the corresponding cases. For example, given a labelled event with its best matched cluster, a represents the number of query-page pairs that are included in both the labelled event and the detected cluster; b represents the number of query-page pairs that are in the detected cluster but not in the labelled event; c represents the number of query-page pairs that are in the labelled event but

	With Evolution			W/O Evolution		
	s_1	s_2	s_3	s'_1	s'_2	s'_3
r (%)	76	68	79	43	39	45
p (%)	91	83	90	66	52	61
$miss$ (%)	24	32	21	57	61	55
f (%)	0.52	1.41	0.82	1.39	1.57	0.93
$Mic F_1$	0.83	0.75	0.84	0.52	0.45	0.52
$Mac F_1$	0.85	0.79	0.87	0.57	0.51	0.56

Table 4: Event detection results.

not in the detected cluster; while d represents the number of query-page pairs that are not included in the detected cluster and the labelled event. Five evaluation measures, *miss*, *false alarm* (f), *recall* (r), *precision* (p), and the F_1 measure (F_1), are defined as follows based on the contingency table.

- $miss = c/(a+c)$ if $(a+c) > 0$, otherwise undefined;
- $f = b/(b+d)$ if $(b+d) > 0$, otherwise undefined;
- $r = a/(a+c)$ if $(a+c) > 0$, otherwise undefined;
- $p = a/(a+b)$ if $(a+b) > 0$, otherwise undefined;
- $F_1 = 2rp/(r+p) = 2a/(2a+b+c)$ if $(2a+b+c) > 0$, otherwise undefined.

To measure the global quality of the event detection results, two average metrics, the micro-average F_1 ($Mic F_1$) and macro-average F_1 ($Mac F_1$) are used [19]. Given n events, the Mic and $Mac F_1$ are defined as follows.

$$Mic F_1 = \frac{2 \times \sum_{i=1}^n a_i}{2 \times \sum_{i=1}^n a_i + \sum_{i=1}^n b_i + \sum_{i=1}^n c_i}$$

$$Mac F_1 = \frac{1}{n} \times \sum_{i=1}^n \frac{2 \times a_i}{2 \times a_i + b_i + c_i}$$

Table 4 shows the quality of the event detection results, where the size of the window in the histogram representation varied from 1 to 3, which is denoted as s_i where i is the window size. By combining the window size and the time granularity, s_1 , s_2 , and s_3 represent the hourly, daily, and weekly interval, respectively. At the same time, the quality of the corresponding event detection results that only use the semantic-based similarity are denoted as s'_i in the table. In general, the event detection results are satisfactory and encouraging. Also, it shows that the window size of the histogram representation may affect the quality of the results. For example, when the window size is set to s_2 , the quality of the results is not as good as others. The reason is that different events may happen and last for different time. For instance, with the window size of s_2 , most of the events that only last for a window size of s_1 cannot be discovered. Moreover, it can be observed that the evolution pattern-based similarity improve the event detection results substantially. Note that in the above approaches only the click-through data is used. We do not analyze the content or structure of the web pages in the click-through data. This indicates that it is indeed possible to detect events using only visitor-centric data.

6.3 Examples of Detected Events

In this section, we present a list of detected events and the corresponding semantics in real world. As the size of the click-through data is prohibitively large, we only take as example a collection 32 days query log that is more than 22 GB. Hence, here we only show some representative events.

Figure 4(a) shows the related queries about the firework event that happened on the US Independence Day (July 4th). It can be observed that the keyword “firework” and related pages are becoming more popular one week before

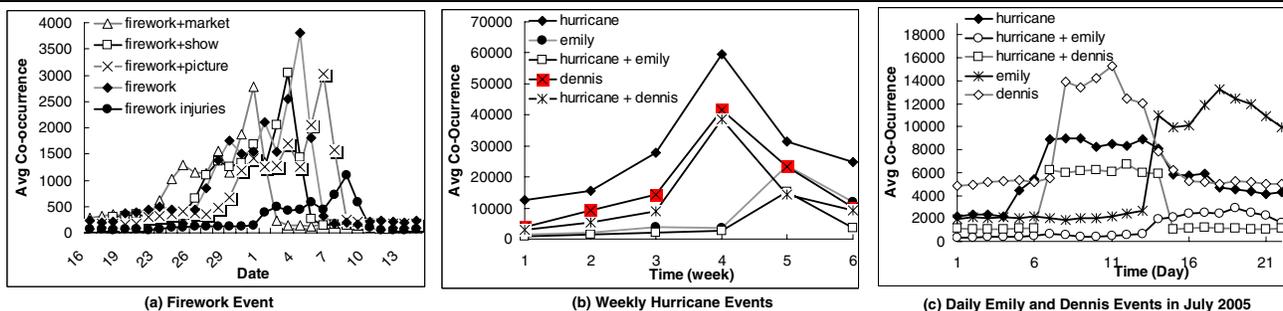


Figure 4: Example of events detected from the real dataset.

the event and reach the peak on July 4th. While other related keywords and web pages such as “firework + buy” and “firework + show” become popular and reach their peaks a few days before July 4th. At the same, the related keywords and web pages such as “firework + injuries” and “firework + picture” have a little delay in terms of the number of times being issued and visited. This observation explains why we use the dynamic time warping-based similarity measure. Note that this event can be most effectively detected based on a daily basis when we vary the window size in the histogram representation.

Figure 4(b) shows the hurricane events that happened in July 2005. It can be observed that the keyword “hurricane” and related pages are becoming more popular during the third, fourth, and fifth weeks starting from June 15, 2005. Observe that related keywords and web pages such as “hurricane + dennis” and “dennis” become popular and reach their peaks in the fourth week. Also, other related keywords and web pages such as “hurricane + emily” and “emily” reached their peak in the fifth week. Note that this experiment is done on a week-based time granularity. In this case, we detected the hurricanes “emily” and “dennis” as a single event because the five queries in the figure have similar evolution patterns. However, the hurricanes “emily” and “dennis” are actually two events that happened within two weeks. If we detect the events on a daily-basis time granularity, then the two events can be successfully separated. This is shown in Figure 4(c) where the five queries have different evolution patterns. More importantly, “hurricane+emily” and “emily” have similar evolution patterns, while “hurricane+dennis” and “dennis” have similar evolution patterns.

7. APPLICATIONS

As the events in our detection results are corresponding to the real world events, they can be used in many applications such as event-level search engine, event alert, event directory, event-based advertisement, etc. Note that the distinguishing advantage our event detection results can bring to these applications is that *event level* is another meaningful level to organize, represent, and retrieve web data besides the web page level using by existing web search engine and the topic level using by existing web directories. In this section, due to the space limitation, we only discuss some of the representative applications.

7.1 Event Alert

With the massive amount of new information that is emerging on the web, it is difficult for common users to find and track most of the materials that they are interested in. Recently, *Google* (<http://www.google.com/alerts>) provides a *news alert* service that can email the users a list of related

news they are interested in based on the specified query terms. The alert basically contains a list of web pages and their snippets in the search results by issuing the query terms to the *Google News*. In this case, results are presented in page-level and ranked in the same way as they are ranked in the search results. However, from our experience, we observed that the alerts may not always be well organized. For example, often there are duplicated or very similar news. Also news about the same event may be ranked quite differently. Using our event detection framework from click-through data, we can provide alerts in the event level and rank them in the event level. Similarly, users can specify their interest using query terms which can be used to provide the list of related events that contains a set of query terms and snippets of the corresponding web pages being clicked. Moreover, based on the evolution pattern, we can classify the event as regular event, periodic event, or burst event and so on.

7.2 Event Directory

In most of the existing web portals, the web directory services are provided. For example, *Google directory* (<http://www.google.com/dirhp>) organized the web by topic into categories. In this case, web pages are organized in a hierarchical manner from broad topics to specific topics. However, such presentation only represents the semantics in the topic level by grouping all similar web pages and web sites, while there may be many different events happening under the same topic over time. These events cannot be distinguished using current directory services techniques. However, an *event directory*, that is build on top of an event detection mechanism, can present the web pages in a more meaningful way. In the event directory, each unit, represented by a group of web pages, is an event rather than a single web page or topic. Similar to the snippets of web pages, snippets of events can also be generated. At the same time, events under the same topic are presented in the descending time order. Also, events can be ordered by categories.

8. RELATED WORK

Recently, a lot of works have been done in the area of topic detection and tracking (TDT) and click-through data analysis. In this section, the differences between our work and existing research are discussed.

The TDT research project was initially a DARPA sponsored project concerned with finding groups of stories from a variety of broadcast news media. It consists of three major issues: segmenting the text corpus into events, tracking the development of the detected events, and detecting new events [18]. Specifically, for event detection, different approaches have been proposed [1, 9, 19]. In [19], a two-phase

novel topic detection algorithm is developed. The idea is to first classify the incoming news into predefined categories and then use the topic-conditioned heuristics to identify the new events. In [1], the authors compare the TDT problem with a more difficult two-part task defined by the TREC 2002 novelty track: given a topic and a group of documents relevant to that topic, 1) find the relevant sentences from the documents, and 2) find the novel sentences from the collection of relevant sentences. In [9], retrospective new event detection (RED) approach is proposed to discovery novel events in the news corpus by taking both the content and the time information.

With the popularity of web search engines, a huge amount of click-through data has been accumulated and available for analysis. Recently, a large body of literature has focused on mining the click-through data for query expansion [3], query and page clustering [15], ranking optimization [6], metadata generation [16], etc. In [3], the authors proposed to extract probabilistic correlations between query terms and document terms by analyzing click-through data. These correlations are then used to select high-quality expansion terms for new queries. In [15], the click-through data is used to improve the content-based query clustering for the application of FAQ identification. In [6], the author proposed to optimize the ranking function of search engines by using the click-through data. The basic idea is to training a ranking function using the SVM with the click-through data. More recently, in [16], Guo et al. proposed an iterative reinforced algorithm to utilize the user click-through data to improve search performance. The algorithm can successfully generate extra metadata for web page by exploring the interrelations between queries and web pages.

Compared to the above mentioned related works, our event detection technique is different in the following aspects. Different from the TDT efforts, this is the first approach that detects events from web log data of search engines *without* analyzing the content and structure of textual data such as web pages. Moreover, unlike existing approaches, the evolution pattern of the click-through data is considered as an integral part of the event detection process in order to reflect the evolution of real world events. Different from the existing click-through data mining research, in our approach we move from the query or page clustering problem, which takes only the semantic relationship between queries and pages, to the query-page pair clustering problem, which combines the evolution patterns of the query-page pairs with their semantic relationships. Such clustering results are expected to represent real world events more accurately.

9. CONCLUSIONS

This work is motivated by the fact that existing event detection approaches only take the author-centric data into account for detecting events, while ignoring the rich collection of visitor-centric data. We believe that the visitor-centric data plays an important role in event detection. In this paper, we proposed a novel approach to detect events from the web by analyzing the click-through data of web search engines. A key feature of our approach is that the dynamic nature of the click-through data is incorporated in the event detection process. In our approach, we model the semantic and evolution pattern-based similarities between query-page pairs in the click-through data as a vector-based graph. Then, the real world event is defined as a set of

query-page pairs that are similar with respect to the semantic and evolution pattern-based similarities. The problem of event detection is then addressed by a two-phase graph cut algorithm on the *dual graph* of the vector-based graph. Experimental results show that our event detection approach can successfully detect many events with high quality.

10. REFERENCES

- [1] J. Allan, C. Wade, and A. Bolivar. Retrieval and novelty detection at the sentence level. In *SIGIR*, 314–321, 2003.
- [2] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *SIGKDD*, 2000.
- [3] H. Cui, J.-R. Wen, J.-Y. Nie, and W.-Y. Ma. Probabilistic query expansion using query logs. In *WWW*, 325–332, 2002.
- [4] F. Harary. *Graph Theory*. Addison-Wesley, 1972.
- [5] G. Jeh and J. Widom. Simrank: a measure of structural-context similarity. In *SIGKDD*, 2002.
- [6] T. Joachims. Optimizing search engines using clickthrough data. In *SIGKDD*, 133–142, 2002.
- [7] E. J. Keogh. Exact indexing of dynamic time warping. In *VLDB*, 406–417, 2002.
- [8] W.-S. Li, K. S. Candan, Q. Vu, and D. Agrawal. Retrieving and organizing web pages by “information unit”. In *WWW*, 230–244, 2001.
- [9] Z. Li, B. Wang, M. Li, and W.-Y. Ma. Retrieval and novelty detection at the sentence level. In *SIGIR*, 2005.
- [10] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE TPAMI*, 22(8):888–905, 2000.
- [11] P. Soundararajan and S. Sarkar. Investigation of measures for grouping by graph partitioning. In *CVPR*, 239–246, 2001.
- [12] A. Sun and E.-P. Lim. Web unit mining: finding and classifying subgraphs of web pages. In *CIKM*, 108–115, 2003.
- [13] J. Sun, H.-J. Zeng, H. Liu, Y.-C. Lu, and Z. Chen. Cubesvd: A novel approach to personalized web search. In *WWW*, 2005.
- [14] Z. Sun, E.-P. Lim, K. Chang, T.-K. Ong, and R. K. Gunaratna. Event-driven document selection for terrorism information extraction. In *ISI*, 37–48, 2005.
- [15] J.-R. Wen, J.-Y. Nie, and H.-J. Zhang. Clustering user queries of a search engine. In *WWW*, 162–168, 2001.
- [16] G.-R. Xue, H.-J. Zeng, Z. Chen, Y. Yu, W.-Y. Ma, W. Xi, and W. Fan. Optimizing web search using web click-through data. In *CIKM*, 118–126, 2004.
- [17] Y. Yang, T. Pierce, and J. Carbonell. A study on retrospective and on-line event detection. In *SIGIR*, 28–36, 1998.
- [18] Y. Yang, J. Carbonell, J. Allan, and J. Yamron. Topic detection and tracking: Detection-task. In *TDT*, 1997.
- [19] Y. Yang, J. Zhang, J. Carbonell, and C. Jin. Topic-conditioned novelty detection. In *SIGKDD*, 688–693, 2002.
- [20] Q. Zhao, C.-H. Hoi, T.-Y. Liu, S. S. Bhowmick, M. R. Lyu, and W.-Y. Ma. Time-Dependent Semantic Similarity Measure of Queries Using Historical Click-Through Data. In *WWW*, 2006.