Int J Digit Libr (2018) 19:3–19
https://doi.org/10.1007/s00799-016-0207-1

Focused crawler for events

Mohamed M. G. Farag[1] · Sunshin Lee[1] · Edward A. Fox[1]

Received: 1 April 2016 / Revised: 27 December 2016 / Accepted: 29 December 2016 / Published online: 7 January 2017

Abstract There is need for an Integrated Event Focused Crawling system to collect Web data about key events. When a disaster or other significant event occurs, many users try to locate the most up-to-date information about that event. Yet, there is little systematic collecting and archiving anywhere of event information. We propose intelligent event focused crawling for automatic event tracking and archiving, ultimately leading to effective access. We developed an event model that can capture key event information, and incorporated that model into a focused crawling algorithm. For the focused crawler to leverage the event model in predicting webpage relevance, we developed a function that measures the similarity between two event representations. We then conducted two series of experiments to evaluate our system about two recent events: California shooting and Brussels attack. The first experiment series evaluated the effectiveness of our proposed event model representation when assessing the relevance of webpages. Our event model-based representation outperformed the baseline method (topic-only); it showed better results in precision, recall, and F1-score with an improvement of 20% in F1-score. The second experiment series evaluated the effectiveness of the event model-based focused crawler for collecting relevant webpages from the WWW. Our event model-based focused crawler outperformed the state-of-the-art baseline focused crawler (best-first); it showed better results in harvest ratio with an average improvement of 40%.

Keywords Event archiving · Focused crawling · Web archiving · Event modeling · Digital libraries

1 Introduction

There is need to improve collection of Web data about key events. Events lead to our most poignant memories. We remember birthdays, graduations, holidays, weddings, and other events that mark stages of our life, as well as the lives of family and friends. As a society we remember assassinations, natural disasters, man-made disasters, political uprisings, terrorist attacks, and wars—as well as elections, heroic acts, sporting events, and other events that shape community, national, and international opinions. Web and social media content describe many of these societal events.

In part, Web 2.0 [1] is a highly responsive sensor of important occurrences in the real world, since people from across the globe meet virtually and share related observations and stories online. We can leverage this stream of data, to trigger event archiving, for automatic collection of event information, and later to enable event-related services that support communities.

Permanent storage and access to big data collections of event-related digital information, including webpages, tweets, images, videos, and sounds, could lead to an important international asset. Regarding that asset, there is need for digital libraries (DLs) [2–4] providing immediate and effective access, and archives with historical collections that aid science and education, as well as studies related to economic, military, or political advantage.

When something notable occurs, many users try to locate the most up-to-date information about that event. Later, researchers, scholars, students, and others seek information about similar events, sometimes for cross-event comparisons or trend analyses. Yet, there is little systematic collecting and archiving anywhere of information about events, except when national or state events are captured as part of government-related Web archives. This is the need addressed

✉ Mohamed M. G. Farag
mmagdy@vt.edu

[1] Virginia Tech, Blacksburg, VA 24061, USA

by the Integrated Digital Event Archive and Library (IDEAL) project [5].

Though the Internet Archive [6] supports some event-oriented archiving, coverage is limited [7]. Many important events are ignored, while others are only captured in part. Further, tools for capture are complex and time consuming, and few archivists master their features, so achieving high recall is expensive. There are few mechanisms to filter out noise in collections. Access to the resulting archives is awkward and inefficient due to the fact that much of the content captured is non-relevant [8]. We argue that manual curation of seed URLs is not scalable and not fully effective for archiving events that have high impact. Thus, improved technology is needed.

The IDEAL project is developing a digital library/archive supporting automatic event tracking, crawling, and archiving, as well as effective access (in the sense of aiding in the finding and utilization of relevant high quality information). By recording tweets, news, webpages, (micro)blogs, and queries, our system collects and archives event-related digital objects and provides a broad range of helpful services.

In this paper, we focus on the data front end of the IDEAL project, i.e., collecting and archiving data using a new type of focused crawler. The IDEAL project has around 11 TB of webpage archives (WARC files) and over 1.2 billion tweets across hundreds of different events [9]. Early on, the webpage archives were collected using the Internet Archives [6] Archive-It service [10], which uses the Heritrix tool for archiving webpages. Originally, the IDEAL project manually prepared a list of URLs for events and fed it to the Archive-It service for crawling. The problem with this weakly curated approach is that we produced collections with low precision (i.e., with few relevant and many non-relevant webpages); Heritrix is a general Web crawler and does not analyze the textual content of the webpages before downloading them. To overcome this problem, the IDEAL team shifted to another approach: We extracted URLs from tweet archives that were built about events, and downloaded only the corresponding webpages. The resulting collections have high precision (most of the webpages were relevant) but low recall (not all of the relevant webpages were found).

A focused crawler would help solve each of the previous problems by crawling (to increase recall) the WWW starting from the URLs extracted from the tweet archives but then following only the relevant webpages (to enhance precision) in order to find and collect as much relevant information as possible. However, in the past, focused crawling was mainly applied to topical crawling (i.e., collecting webpages about a certain topic or domain). Accordingly, since we are working with events, we extended the approach previously used by the IDEAL team and adapted/changed the traditional focused crawler approach to accommodate our needs.

We proposed two major changes to traditional topical focused crawling: (1) a novel event model and representation and (2) incorporating the event model information extracted from seed webpages content into focused crawling. We show that the extracted event model information increases focused crawler effectiveness and helps identify more of the relevant webpages while maintaining high levels of precision.

We conducted two series of experiments. In the first experiment series, we evaluated the effectiveness of the event model to help classify webpages into relevant and non-relevant classes. In the second experiment series, we evaluated the effectiveness of the event model-based focused crawler to crawl and collect relevant webpages, showing it to be better than the baseline topical crawler.

Our contributions in this paper are:

1. A model and representation for capturing the different aspects of events in webpages (topic, location, and date);
2. An extended focused crawler approach that uses our event model to represent content and to estimate the relevance of webpages.

The rest of this paper is organized as follows: Sect. 2 reviews different approaches for focused crawling. Section 3 describes the architecture of the baseline focused crawler. In Sect. 4, we propose our new event model and representation, and explain how it is integrated with the focused crawling approach. In Sect. 5, we discuss the design of our experiments, datasets used, and the evaluation measures. Section 6 presents an evaluation of our event model-based focused crawler and of the baseline focused crawler. Finally, Sect. 7 concludes and discusses issues for future research.

2 Related work

Most of the work done in traditional topical focused crawling [11] falls into one of three categories: machine learning, semantic similarity, or content and link analysis. We discuss the major work done in these three categories in the next subsections. Along the way, we also touch on publications related to event modeling.

2.1 Machine learning

Machine learning-based focused crawler approaches apply text classification algorithms [12,13] to learn a model from training data. The focused crawler then uses the model to estimate the relevance of unvisited webpages. The use of the model enhances the performance of the classifier by incorporating domain-specific knowledge and online relevance feedback. Although our approach can be considered as involving a classification task, we do not require training data (i.e., we instead directly use positive examples), and we are using webpage text for building the event model, not for building a classifier.

Rennie and Barto [14] used reinforcement learning for solving the focused crawling problem. They modeled the focused crawling problem as a Markov decision process with webpages as states, URLs as actions, and on-topic webpages as the rewards. Another reinforcement learning algorithm, temporal difference learning, was used in [15]. They used a state value function to estimate the importance of webpages to lead to future relevant webpages. A later work [16] used the reinforcement learning framework proposed in [14] and enhanced its performance by applying incremental online learning. For each new URL, they identify its corresponding class and use its features to update the class features and q-value. They retrain the supervised learning algorithm based on the new training data. This approach eliminates the data bias that appears in the test data, where unseen URLs may appear from new domains that were not found in the training data.

Infospiders is a topical crawler based on adaptive online agents [17] that use genetic programming and reinforcement learning approaches to estimate the relevance of a webpage. In our approach, we go beyond just using topics and use event modeling for estimating the relevance of a webpage and to adapt to changes in event topic.

2.2 Semantic similarity

Semantic similarity-based techniques use ontologies [18, 19] for describing the domain of interest. The domain ontology can be built manually by domain experts, or automatically, using concept extraction algorithms. Once the ontology is built, it can be used for estimating the relevance of unvisited webpages by comparing the concepts extracted from the target webpage with the concepts that exist in the ontology. The performance of semantic focused crawling depends on how well the ontology describes and covers the domain of interest.

We are using disaster domain knowledge for finding disaster-specific keywords. Our event model could be easily mapped to portions of our event ontology, but further research would be needed to see whether such would yield improvement. Further, using this approach, in a system that is aimed to handle any type of important event, would require considerable knowledge engineering work.

2.3 Content and link analysis

Text and link analysis algorithms combine text analysis schemes (e.g., vector space model) and link analysis algorithms to estimate the relevance and importance of webpages [20–22]. Link analysis approaches introduce the concept of popular webpages, measured based on the link structure of the WWW. This led to the introduction of the concept of Hub and Authority webpages [23]. Hub webpages have links to many authority webpages while authority webpages are linked from many hub webpages. Among the link analysis algorithms, Page-Rank [24] is the most used. Alternatively, context graphs are used to represent the context of a webpage using neighborhood webpages that are most similar to it [22].

Another line of research incorporates the genre of webpages into focused crawling [25]. The genre of the webpage defines the type of the webpage (e.g., forum, tutorial, news, blog, course-syllabus). The focused crawler uses two sets of terms, one for determining the genre of the webpage and the other set for determining the topic. The two sets of keywords (genre and topic) are manually determined by experts and then used by the focused crawler for estimating the relevance of the webpage.

Pant et al. [26, 27] describe a new Web characteristic, status locality on the Web. A webpage's status measures the importance of the webpage with respect to its popularity and is approximated by the number of links pointing to it. Pant et al. developed an algorithm for estimating the status of a webpage based on local characteristics of the webpage and also demonstrated that the status property has some of the same characteristics as the topical property.

Chen [28] developed a hybrid approach for focused crawling using genetic programming for exploiting different features in a webpage's text, and metadata search for exploring different sources on the WWW. Chen tried to solve the exploration versus exploitation trade-off for focused crawling. He tackled the exploitation task by using the genetic programming approach for combining different relevance signals from the webpage text. For the exploration task, he used metadata search for gathering several seed URLs for the crawler to start from, thus expanding the crawler coverage of the WWW. In order to overcome the bias that can be found in one search engine, he used multiple search engines and combined their results.

Event modeling recently has attracted interest in different fields, like topic detection and tracking (TDT) [29], animal disease outbreak detection [30], networked multimedia events [31], and document similarity [32]. In [29], an event is described as a topic that happens at a certain time, in a specific location, and with a particular set of participants. In [31], an event is defined as a tuple of aspects: informational, spatial, temporal, structural, causal, and experiential. The informational aspect includes event ID, type, and other attributes. Spatial and temporal aspects represent the location and time properties, respectively. The structural aspect includes the sub-events belonging to the current event. The causal aspect includes the events causing the current event. Finally, the experiential aspect includes all media resources related to the current event. We incorporated similar event modeling into our crawler, to build an event-aware focused crawler [33]. We came up with a simple event model that integrates ideas from TDT [34] and work done in [30]; here, we also report on its experimental validation.

In [35], the authors combined the focused crawler technique with social media to improve the freshness of the crawl. A focused crawler is limited by the set of seed URLs it starts from. Social media produce a huge amount of user-generated content (e.g., tweets) that may contain URLs. Since social media content is produced live, the URLs contained therein would be fresh and possibly more recent than the URLs visited by the focused crawler. Injecting URLs from social media into the focused crawler's URLs queue (frontier) should increase the freshness of the Web collection produced. The authors crawled about two events—Ebola and Ukraine conflict—and used a keyword-based model to represent the two events.

In [36], the authors examined the topical quality of existing Web archives about events. They built a framework that assesses whether the seed URLs used in building the Web archive are on-topic or off-topic across the different times it was crawled. The authors used the vector space model (VSM) [37] to represent the documents and applied several measures to calculate the similarity scores. They evaluated their method using different thresholds to find the value that yielded the best performance.

3 Focused crawling

3.1 Crawler architecture

A general Web crawler consists of: webpage fetcher (downloader) for retrieving webpage contents, frontier for storing unvisited URLs, and webpage processor for extracting text and URLs out of a webpage's HTML. Crawlers model the WWW as a graph $G(V, E)$ where nodes (V) are webpages and edges (E) are links between webpages. So, two webpages (nodes) will have an edge between them if one webpage has a link pointing to the other webpage.

Similar to general Web crawling, a focused crawler has a webpage fetcher, frontier, and webpage processor. In addition, a focused crawler has a topic or domain-specific model and a module for estimating the relevance of URLs and webpages. Typically, a focused crawler takes as input: (1) the desired number of pages to collect and (2) seed URLs to start crawling from. It outputs the set of webpages found.

One of the important aspects of focused crawlers is the ordering of the URLs in the queue, which specifies the order of visiting the nodes of the graph. In the focused crawler literature [11,38], best-first search is the most commonly used technique and is considered the state-of-the-art focused crawler, taking into consideration the estimated relevance of the URLs/webpages during crawling.

A focused crawler starts from a seed URL [11]. It downloads the corresponding webpage and extracts the text of that webpage. The focused crawler then estimates the relevance of the webpage textual content with regard to the topic/event of interest. In the next step, there are two design options. One approach is for the focused crawler to decide whether the webpage is relevant or not by comparing its estimated score to a predefined threshold. If the webpage is considered relevant, then the focused crawler extracts the embedded URLs from the webpage and inserts them into the frontier. The other approach is for the focused crawler to extract all embedded URLs from the webpage and then insert those into the frontier, not considering the score. The second option takes into consideration the tunneling phenomena in crawling, where a non-relevant webpage links to relevant webpages, either directly or through several steps.

When inserting the extracted URLs into the frontier, the focused crawler has to make another decision. One option is to insert all extracted URLs, along with the estimated score of the webpage from which they were extracted. Another option is to estimate the relevance of each URL based on the characters in the URL and insert the URL and the resulting score into the frontier. A hybrid approach (the option we adopted) is to use the average of a URL's score and the score of the webpage from which it was extracted.

Algorithm 1: Baseline Focused Crawler

```
Input: Seed URLs, numPages, urlScoreThreshold
Output: crawled webpages
Insert URLs in Priority Queue;
Download seed webpages from seed URLs;
topicVec $\leftarrow$ Build topic representation from seed webpages;
while $pagesCount < numPages$ do
    URL $\leftarrow$ pop(priorityQueue);
    append URL to visited list;
    webpage $\leftarrow$ download(URL);
    webpageVector = process(webpage);
    pageScore = calculateScore(webpageVector, topicVector);
    pagesCount +=1;
    webpageOutgoingURLs $\leftarrow$ extract outgoing URLs from
    webpage;
    add webpage to saved collection;
    for link in webpageOutgoingURLs do
        validate(URL);
        if URL not in visited list and URL not in priorityQueue
        then
            urlVector $\leftarrow$ process(URL text);
            urlScore $\leftarrow$ calculateScore(urlVector, topicVector);
            if $urlScore >= urlScoreThreshold$ then
                push(URL,priorityQueue);
            end
        end
    end
end
```

Finally, the focused crawler pulls from its frontier the URL with highest score and repeats the process. Algorithm 1 shows a focused crawler algorithm that handles tunneling (i.e., extracts the URLs from the webpage regardless of score): estimating the score of each URL and inserting it into the queue with that score. We consider this approach as the foundation for the baseline for evaluation comparisons discussed below.

3.2 Topic representation

One of the inputs to a focused crawler is a set of URLs; together these can be used to describe the event/topic of interest (model). This set of URLs is selected on the basis of providing high-quality textual content about the event/topic. The focused crawler uses this set of URLs to build its event/topic model and then uses the model to estimate the relevance of the URLs and webpages it encounters during crawling.

In the rest of this paper, we consider two ways to represent an event. In the rest of this subsection, we consider the first, traditional, baseline approach, where an event is treated like a topic, characterized by a set of keywords. In Sect. 4, we describe our new approach, where an event is described with a richer model.

Our baseline best-first search focused crawler uses the vector space model (VSM) [37] approach to build its event/topic model:

1. Using the model URLs, download corresponding webpages and extract text from those webpages. Each document is tokenized to a set of words—stop words removed, and words stemmed—and then converted to a vector, where the vector represents the unique words in the documents and their frequencies (how many times they appeared in the documents).
2. The crawler then builds a vocabulary index using the webpages' vectors for the seed URLs. The vocabulary index maps the set of unique words in all the webpages to a list of the word frequencies in the webpages.
3. Using the vocabulary index, the crawler calculates a weight for each word by summing all its frequencies in the webpages in which it appeared. This corresponds to the word collection frequency, which is the sum of the word Term Frequency (TF) values over all webpages. We haven't used the word Inverse Document Frequency (IDF) as we are using the seed webpages only, not a general corpus; in this situation, possibly discriminative features that appear in all seed webpages would have very small (or even zero) weight if we tried to use IDF.
4. The crawler selects the top k words with highest weights as a model for the event/topic of interest. The weights of the words are calculated by using the log of the frequencies, to reduce the effect of long webpages dominating short documents.

The baseline crawler uses the event/topic model to represent the event/topic of interest and also to model each webpage it visits during crawling. After getting a URL with highest score from the frontier queue, the crawler downloads the webpage, extracts the text, and then converts the text to a vector of words with their frequencies according to the vocabulary built from seed webpages. The crawler then estimates the relevance of the webpage by calculating the cosine similarity between the event/topic vector and the webpage vector.

Also, the crawler estimates the scores of all the URLs in that webpage. For each URL, the crawler combines the URL tokens and anchor text, converts them to a vector of features with their frequencies, and calculates the cosine similarity of the URL vector to the event/topic vector. The URL will be inserted into the frontier with the estimated score. The vocabulary (keywords or other features) which the crawler uses to represent the webpage vectors and the extracted URL vectors is built and extracted from the set of seed URLs. The webpage vector is used to estimate the relevance of the webpage and produce a relevance score. The URL score is calculated as an average of the URL vector score (calculated as the cosine similarity between topic vector and URL vector) and the score of the webpage in which the URL appeared (see previous section for details). So, the crawler is making use of three types of textual information: webpage text, URL anchor text, and URL address tokens. Using webpage text and URL information was proved to be more efficient than using webpage text only [11,39].

Figure 1 shows the architecture of the baseline best-first focused crawler with the topic representation and relevance estimation processes highlighted in dashed boxes.

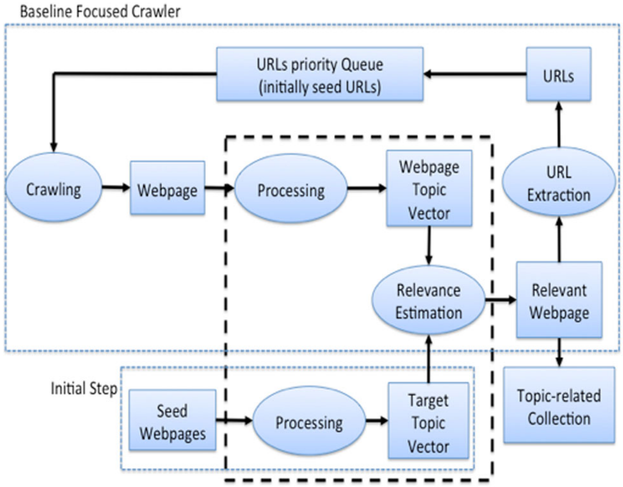


Fig. 1 Baseline focused crawler architecture

4 Event model-based focused crawling

4.1 Event model and representation

In the focused crawler literature [35], events are considered as topics and are represented with a list of keywords. Although this approach might work well for some events, it works less well for other kinds of events. For example, representing events as a list of keywords would work in cases where the topic part of the event is most dominant and important, while the location and time parts are not important or do not play a significant role in the event. The outbreak of Ebola is a good example of an event where the topic (spreading of Ebola) is the most important aspect. The location and date are part of the details, but are not that important, i.e., the topic part is sufficient to clearly describe the event. On the other hand, shooting events, for example, cannot be described with the topic part only. Since there are many shooting events in different places and at different times, we need the location and date parts to clearly describe a particular shooting event.

Event Model: Before considering complex event models, a simple scheme should be tested first. Thus, we define an event as something (e.g., a disaster), which happened in a certain place, and at a certain time. Thus, an event E is a tuple $<T, L, D>$. The three parts reflect what, where, and when. Thus, T is the topic of the event, L is its location, and D is its date. These are explained below in more detail.

Topic: Using a set of seed URLs, we create an event vocabulary (a set of unique keywords that appear frequently in the webpages associated with those seeds). We represent an event with a reference vector created by taking the top m keywords from the event vocabulary.

Date: The event date is given by a user or is extracted automatically from the set of seed webpages. The event date represents the starting date when the event first occurred. The event also could have an ending date.

Location: A small set of location entities is likely to appear frequently in seed webpages, representing places related to the event. These location entities are extracted (as described next) from seed webpages' text; we perform a frequency analysis to help find the most important location entities mentioned in the seed webpages.

For example, we model the shooting that happened in San Bernardino, California, on December 2, 2015, as follows:

- Topic: shooting, shooter,..., etc.
- Location: San Bernardino, California
- Date: 12/02/2015

Similarly, we model the attack that happened in Brussels, Belgium, on March 22, 2016, as follows:

- Topic: terror, attack, explosion,…, etc.
- Location: Brussels, Belgium
- Date: 3/22/2016

Our event model (combining topic, location, and date) can default to the topic-only model in the case of Ebola by ignoring the date and location part (by setting the weights of the location and date parts to zero). We note that this would be the case also regarding the Zika virus outbreak. We can add a part in our system that if the event type is disease outbreak (manually entered by the user), the system automatically defaults to the topic-only model.

Thus, our system is flexible and can be used in cases where it is difficult to determine, for a given event, which model is more efficient. But if the event is a news/world event which is physically localized (has a clear center) and temporally limited (with an impulse in number of documents published in a relatively short period), then we believe our event model (combining topic, location, and date) will perform more efficiently than the topic-only model. Thus, if a user is interested in the outbreak of Zika/Ebola in a certain place and at certain time, then our event model (combining topic, location, and date) should perform better than the topic-only model.

Figure 2 shows the steps of building an event model from seed webpages. We start the process of building the event model by downloading the webpages corresponding to the seed URLs. We then extract the date from the seed URLs and the seed webpages. To do this, we first try to extract the publication date from the seed URLs using a predefined regular expression. If that fails, we extract the publication date by parsing a predefined set of tags from the webpages.

For the date extraction, we have used a library for extracting the publishing date of a webpage using heuristics. The first step is to extract the publishing date from the URL using a regular expression, if applicable. For example, the URL http://www.cnn.com/2016/07/10/us/black-lives-matter-protests/index.html has a publishing date of July 10, 2016. If the URL doesn't contain date information, then next step is to look for specific tags in the header part of the corresponding webpage HTML tags. An example tag that contains publishing date looks like:

<meta name="pubdate" content="2015-11-26">

This appears in the head tag of a webpage. There are multiple metatags that might contain publishing date; hence, the library has an extensive list of possible metatags that are frequently used in websites. The final step is to check in the body of the webpage, if no publishing date is found in the head tag. As before, a list of frequently used body tags is used to guide finding the publishing date. An example of such a tag containing publishing date is:

<p class="pubdate"> Sept 3, 2011 </p>

If there are multiples dates found in the webpage, the library returns the first one only. The library tries to extract the pub-

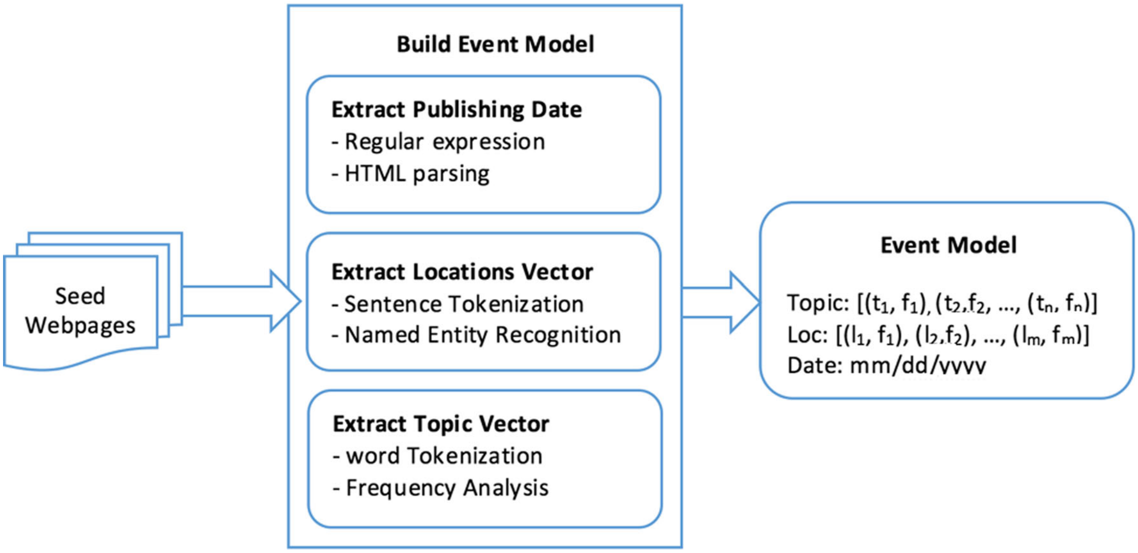


Fig. 2 Steps for building event model from seed webpages

lishing date from the URL, then from the head tag part of the HTML, and then from the body part of the HTML. The order is important, since a date extracted from the URL is expected to be more accurate than from the head than from the body.

An extra step that could be done is to use natural language processing techniques to extract named entities (i.e., dates) from the textual content of the webpage. Using such an extracted date, we can figure out the publishing date of the webpage. However, we have not used this approach, because with the library we managed to extract publishing dates from most of the websites, and because of the overhead of using a named entity recognizer.

For the event model locations vector, we segment the text of the webpages of the seed URLs into sentences and apply Stanford Named Entity Recognition (SNER) [40] on each sentence to extract location entities. We then perform frequency analysis on the extracted location entities and construct the locations vector. It includes the unique locations extracted, along with their frequency of occurrence in all sentences in all seed webpages (i.e., the weight of each location is the cumulative frequency in all seed webpages).

The resulting locations vector will include the frequent locations mentioned in the set of seed URL webpages, which should be the location of the event of interest, assuming those webpages are relevant and of high quality (with regard to containing enough information about the different event aspects, namely topic, location, and date). The SNER could extract location entities not related to the event from some of the seed webpages, as a webpage may include references to multiple locations. This should not affect the model, however, as the frequency of those location entities should be very small (since they typically appear in few of the webpages). On the other hand, if the event occurs in multiple locations, a suitable list of locations should be found through the above-mentioned processing of webpages for the seed URLs.

For the topic vector, we perform the same process as for the baseline VSM: We tokenize the text of the webpages of the seed URLs into words, remove stop words, stem word forms, perform frequency analysis, and construct the topic vector, formed from the set of unique words along with their frequency of occurrence.

4.2 Event model-based webpage and URL scoring

In this section, we show how the focused crawler uses the event model to calculate a score for each webpage it visits and for the URLs extracted from those webpages.

4.2.1 Calculating weights

The weights a, b, and c could be set manually by an expert who would take into consideration the type of the event (shooting, hurricane, bombing, earthquake, etc.) and the characteristics of the event (time duration and location area, i.e., specific location and point in time for a "sharp" event, versus multiple locations and long time periods for complex events).

To automatically calculate the weights, we use each aspect of the event model (topic, location, and date) separately to score a sample of labeled webpages (i.e., relevant and non-relevant webpages). We evaluate each aspect's performance (cosine similarity score) using different threshold values, and we choose the threshold value that produces the best classification performance according to a given evaluation metric.

We used the F1-score as the classification evaluation metric [41]. F1-score is an information retrieval metric that calculates the geometric mean of the precision and recall. We also used the F1-score to assign the weight of each aspect of the event model (topic, location, and date) which indicates the importance of that aspect in calculating the final score. We calculate the weight as the ratio of the aspects F1-score to the sum of the F1-scores of all aspects.

For example, let us assume we have 100 webpages, 50 relevant and 50 non-relevant to a specific event (California shooting, for example). Assume also we have the target event model (extracted from another set of relevant webpages or entered manually by the user). For each webpage of the 100 webpages, we extract the topic vector, locations vector, and the publication date. Then we calculate three scores (topic score, location score, and date score) for each webpage using Equations 1, 2, and 3, respectively. After this process, we end up with a matrix of 100 rows (webpages) and 3 columns (topic score, location score, and date score). Now, we use each of the scores (topic, location, and date) separately to predict a label (relevant or non-relevant) for each webpage. We produce the label by comparing the score to a threshold value (call it K); we predict relevant if the score is larger than the threshold and non-relevant if it is smaller.

After this process, we end up with a matrix of 100 rows (webpages) and 3 columns (label based on topic score, label based on location score, and label based on date score). Then we evaluate the effectiveness of each aspect of the event (topic, location, and date) by comparing the actual labels and predicted labels for each of the three aspects (topic, location, and date). We used the F1-score as the metric for evaluation. Here we end up with 3 F1-scores (one for each of the topic, location, and date) for the threshold value K. We repeat the previous process for different values (n values) of the threshold parameter. Then we end up with a matrix of n rows (different values of the threshold) and 3 columns (topic, location, and date). Finally, we choose the max F1-score for each aspect (topic, location, and date). The weight of each aspect will be the ratio of its F1-score to the sum of three aspects F1-scores.

The weights of each aspect of the event model (topic, location, and date) are learned before the crawling time by applying the above-mentioned procedure on a given set of URLs and webpages that are labeled as relevant or non-relevant. The weights learned are used during crawling and are not modified.

4.2.2 Webpage scoring

A focused crawler gives each webpage it downloads a score, which estimates the relevance of the webpage to the event. It considers event aspects during the relevance estimation process. In particular, we score the relevance of a webpage with respect to each aspect of the event and then combine the partial scores into a final score.

According to our event model, there are three attributes which together fully describe an event. A webpage can have some or all of the attributes of an event. A webpage is considered relevant (i.e., talks about the target event) if it satisfies the following conditions:

- It has a non-empty subset of the keywords that represent the topic attribute of the target event (i.e., is topically relevant).
- Its publication date is close to the event date.
- It has a non-empty subset of the locations attribute of the target event (i.e., the location entities extracted from the webpage are similar to event location entities).

A webpage that satisfies these conditions should be considered relevant and will be added to the output collection.

The event focused crawler first takes the following steps with regard to a webpage:

1. Extract the text of the webpage.
2. Extract the publication date of the webpage.
3. Extract location entities from the text of the webpage (using Named Entity Recognition, NER).

We have developed a function to measure the similarity between the target event model and the webpage model. The similarity function produces a score that estimates the relevance of the webpage to the target event.

Given a target event model and a webpage event model: $e_1 = (T_1, L_1, D_1)$ and $e_2 = (T_2, L_2, D_2)$, where T_1 is the event reference topic vector, L_1 is the vector of location entities extracted from seed webpages using SNER, D_1 is the event date, T_2 is the vector representation of the webpage text, L_2 is the vector of location entities extracted from the webpage text using SNER, D_2 is the publication date of the webpage, e_1 is the target event model, and e_2 is the webpage event model.

The similarity function $sim(e_1, e_2)$ is defined as:

$$sim(e_1, e_2) = a \times sc(T_1, T_2) + b \times sc(L_1, L_2) + c \times sc(D_1, D_2) \tag{1}$$

where

$$sc(T_1, T_2) = \frac{\sum_{t \epsilon T_1 \cap T_2} w(t_1) \times w(t_2)}{|T_1| \times |T_2|} \tag{2}$$

i.e., the cosine similarity between the T_1 and T_2 vectors and where $w(t_i)$ is the weight of term t in document i, and

$$sc(L_1, L_2) = \frac{\sum_{l \epsilon L_1 \cap L_2} w(l_1) \times w(l_2)}{|L_1| \times |L_2|} \tag{3}$$

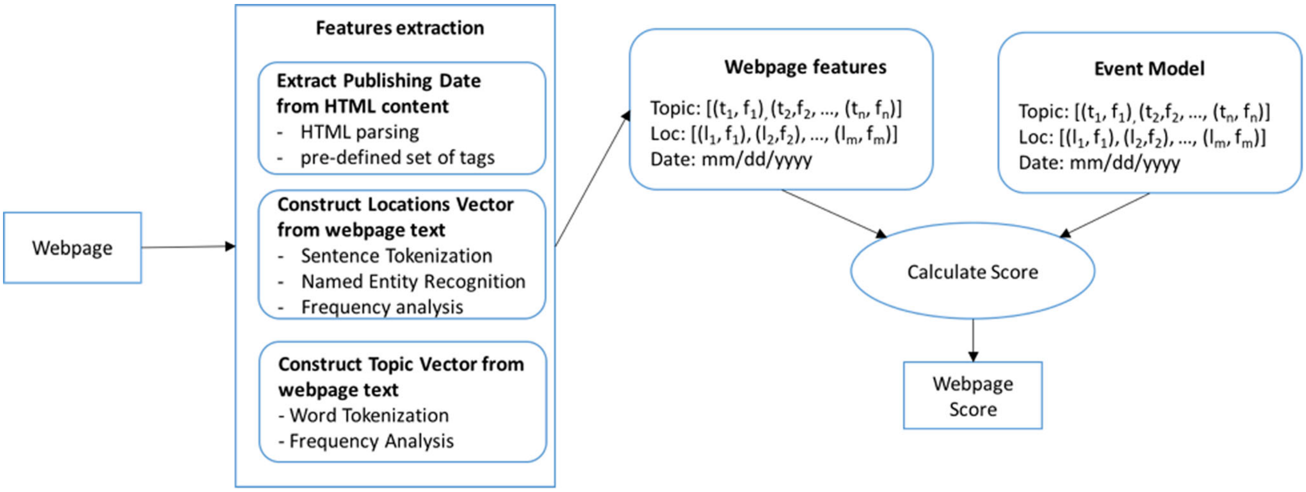

Webpage
Features extraction
Extract Publishing Date from HTML content
- HTML parsing
- pre-defined set of tags
Construct Locations Vector from webpage text
- Sentence Tokenization
- Named Entity Recognition
- Frequency analysis
Construct Topic Vector from webpage text
- Word Tokenization
- Frequency Analysis
Webpage features
Topic: $[(t_1, f_1),(t_2,f_2, \ldots, (t_n, f_n)]$
Loc: $[(l_1, f_1), (l_2,f_2), \ldots, (l_m, f_m)]$
Date: mm/dd/yyyy
Event Model
Topic: $[(t_1, f_1),(t_2,f_2, \ldots, (t_n, f_n)]$
Loc: $[(l_1, f_1), (l_2,f_2), \ldots, (l_m, f_m)]$
Date: mm/dd/yyyy
Calculate Score
Webpage Score

Fig. 3 Steps for calculating the score of a webpage

i.e., the cosine similarity between the L_1 and L_2 vectors and where $w(l_i)$ is the weight of location l in document i, and

$$sc(D_1, D_2) = 1 - \frac{|D_1 - D_2|}{\textbf{numDays}} \tag{4}$$

where **numDays** is the number of days in a year.

The final score of the webpage is calculated by using a weighted average of the scores of the topic, location, and date vectors, where constants a, b, and c are the weights of topic, location, and date scores, respectively. These add to one: $a + b + c = 1$. Figure 3 shows the steps for calculating the score of a webpage.

4.2.3 URL scoring

A similar procedure to that for webpages is implemented for estimating a score for each URL extracted from a webpage. A URL is converted into tokens by removing non-alphabetic characters (like /, # , and ?) and also removing URL-specific keywords like (‘http’, ‘com’, ‘www’). URL tokens are combined with tokens from associated anchor texts. The resulting tokens are then converted to a bag-of-words-based vector representation. We extract the location entities from URL anchor text using SNER and extract the publication date from the URL using regular expressions (if applicable). Figure 4 shows an example webpage about the Brussels attack with a relevant URL highlighted. The anchor text of the URL is: “Paris and Brussels terror suspect to face charges in France.” The address that the URL points to is: https://www.theguardian.com/world/2016/jun/09/mohamed-abrini-paris-brussels-terror-suspect-france-man-in-the-hat. We can see that the URL’s address contains the publication date of the corresponding webpage (June 9, 2016). The URL’s vector after tokenization, stop word removal and stemming would be based on:

```
[‘guardian’,‘2016’,‘jun’,‘mohamed’,‘
abrini’,‘paris’, ‘brussels’,‘terror’,
‘suspect’, ‘france’,‘man’, ‘hat’,
‘charge’].
```

We note that SNER will capture the location entities from the URL’s anchor text only, not the URL address, because the SNER segments its input into sentences and tries to extract entities from these sentences. It is straightforward to do so with anchor text but generally infeasible with a URL (since a set of URL tokens is rarely a meaningful sentence).

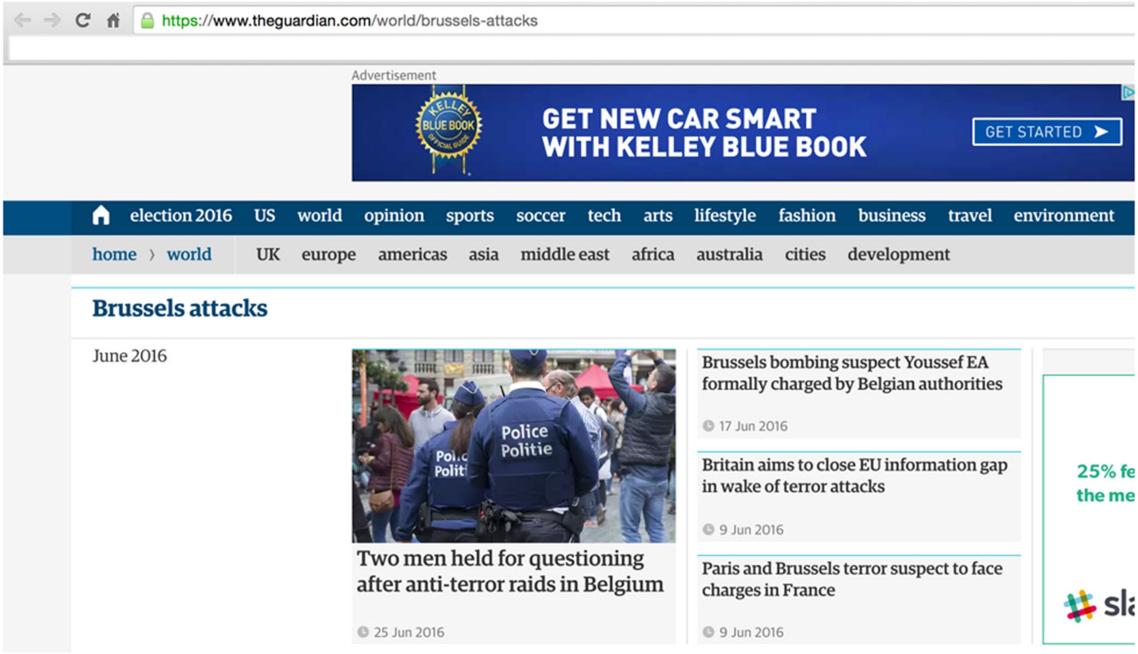
If the set of seed URLs is from one domain (e.g., www.theguardian.com), this may effect the quality of the information extracted to build our event model. The coverage from a single domain could be biased or limited in scope, while that is less likely if there are multiple domains. We estimate the relevance score of the URL using the same procedures used for webpages, as described above. Figure 5 shows the steps for calculating the score of a URL.

5 Experimental setup

In this section, we describe the dataset used for our evaluation (Sect. 5.1), the different experiments performed (Sect. 5.2), and the evaluation metrics (Sect. 5.3). We performed two series of experiments, the first to see whether our event model is useful in classification and the second to see whether the event model is useful in focused crawling.

5.1 Dataset

For the focused crawling experiments, we considered two events: the shooting in San Bernardino, California, on December 2, 2015, and the terrorist attack in Brussels on



Brussels attacks

June 2016

Two men held for questioning after anti-terror raids in Belgium

25 Jun 2016

Brussels bombing suspect Youssef EA formally charged by Belgian authorities

17 Jun 2016

Britain aims to close EU information gap in wake of terror attacks

9 Jun 2016

Paris and Brussels terror suspect to face charges in France

9 Jun 2016

Fig. 4 An example webpage with a relevant URL anchor text highlighted

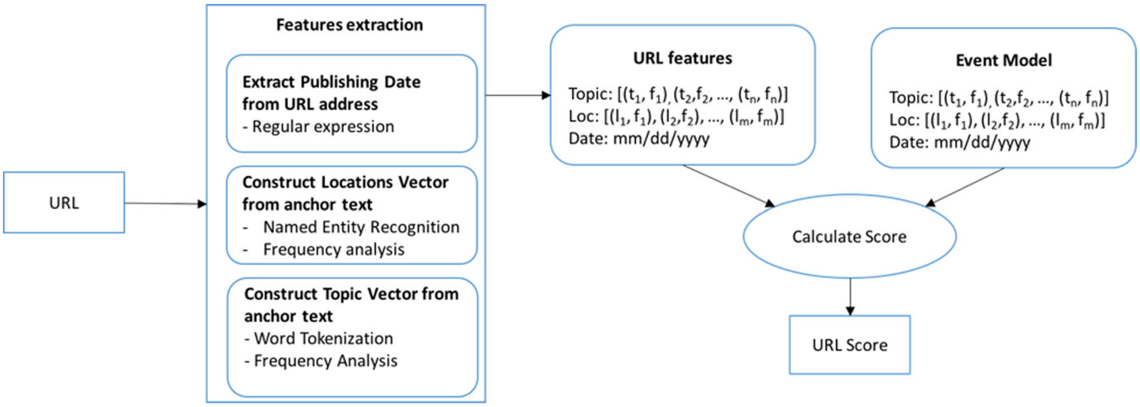

Features extraction
Extract Publishing Date from URL address
- Regular expression
Construct Locations Vector from anchor text
- Named Entity Recognition
- Frequency analysis
Construct Topic Vector from anchor text
- Word Tokenization
- Frequency Analysis
URL
URL features
Topic: $[(t_1, f_1),(t_2,f_2, \ldots, (t_n, f_n)]$
Loc: $[(l_1, f_1), (l_2,f_2), \ldots, (l_m, f_m)]$
Date: mm/dd/yyyy
Event Model
Topic: $[(t_1, f_1),(t_2,f_2, \ldots, (t_n, f_n)]$
Loc: $[(l_1, f_1), (l_2,f_2), \ldots, (l_m, f_m)]$
Date: mm/dd/yyyy
Calculate Score
URL Score

Fig. 5 Steps for calculating the score of URL

March 22, 2016. For the classification experiments, however, it seemed sufficient to just consider the California shooting.

Since the IDEAL project is working with a large amount of event-related data, and since we wanted our results to be assessed in the context of such types of data, we had ample opportunity to utilize data from events like those mentioned above. In the literature [11,13,38], the most used dataset for evaluating topical focused crawlers is the DMOZ dataset. But since we are evaluating focused crawlers for events, the DMOZ dataset is not suitable in our case. Accordingly, we manually curated two sets of URLs about the two events (38 for California shooting and 23 for Brussels attack); these can be used as seeds for the focused crawlers.

For the first series of experiments, about classification, we devised a static dataset for the evaluation of our event model, as is commonly done. The dataset, tailored for studying about classification regarding the California shooting, consists of 1000 URLs and the corresponding webpages. Since there was no existing dataset about that shooting event, we used a keyword-based focused crawler to fetch the 1000 webpages. We used the 38 manually curated URLs as seeds, and the two words “California” and “shooting” as keywords for the crawler. We manually labeled the webpages into two classes (relevant and non-relevant). There were 725 webpages labeled as relevant and 275 labeled as non-relevant.

5.2 Experiments

As mentioned, we performed two series of experiments. The goal of the *first series* was to validate that our event model

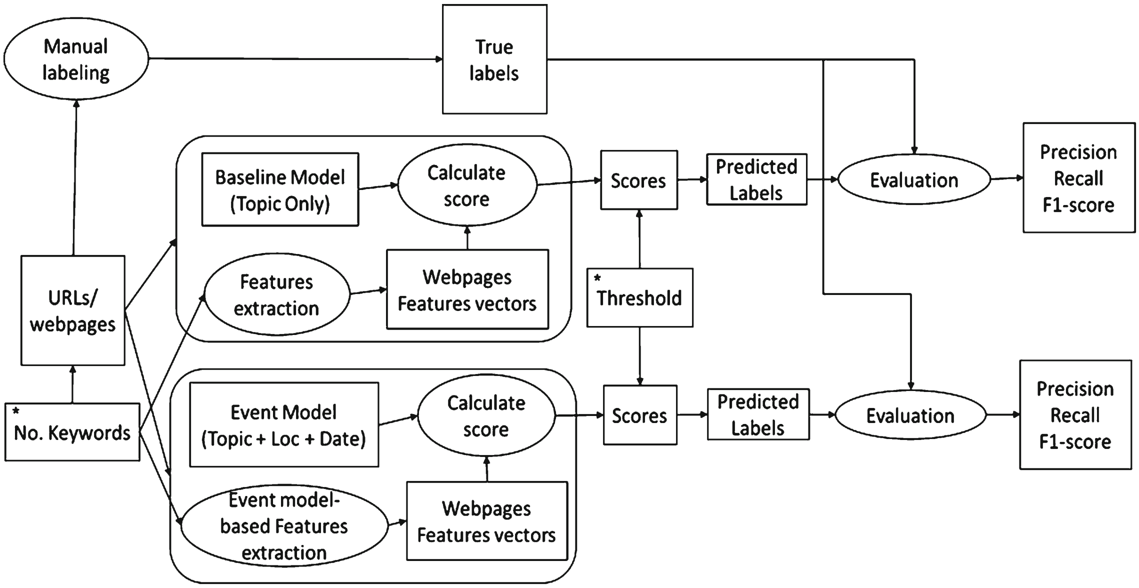

Manual labeling
True labels
Baseline Model (Topic Only)
Calculate score
Features extraction
Webpages Features vectors
Scores
Predicted Labels
Evaluation
Precision Recall F1-score
* Threshold
URLs/ webpages
* No. Keywords
Event Model (Topic + Loc + Date)
Calculate score
Event model-based Features extraction
Webpages Features vectors
Scores
Predicted Labels
Evaluation
Precision Recall F1-score

Fig. 6 Design of our evaluation method of the effectiveness of the event model for relevance estimation; the *two boxes with asterisk* are the two parameters optimized in the experiment

can effectively classify webpages with regard to relevance to the event of interest. We compared our approach against the baseline, using the traditional vector space model (VSM) topic-only approach. We used the manually curated seed URLs and the static dataset of 1000 webpages about the California shooting for the evaluation. The seed URLs are used to build the event model for our approach, and the topic reference vector alone is used for the VSM approach. We used the event model and the topic reference vector to score the 1000 webpages by converting the webpages to their corresponding keyword vectors, and used cosine similarity to calculate the score. We evaluated the performance by varying two parameters:

1. k, the number of keywords used in constructing the topic vector in our event model and the topic reference vector for VSM, and
2. threshold, the value of the threshold used for converting the scores to labels (relevant if the score is larger than the threshold, otherwise non-relevant).

We also ran the experiments with several variations of our event model. We then ran the same experiment with the two pairs of feature types: (a) combination of the topic and location only and (b) combination of the topic and date only. Figure 6 shows the design of the experiments for evaluating the effectiveness of classification using the event model.

In the *second series* of experiments, we aimed to validate that the event model can effectively estimate the scores of the URLs and webpages it visits and consequently guide the focused crawling process to webpages relevant to the event of interest. In these experiments, we used both of the events being studied.

5.3 Evaluation metrics

In the first series of experiments, we used the F1-score metric to evaluate the classification performance of our event model (topic, location, and date) versus the baseline (topic-only).

For the second series of experiments, when evaluating the performance of the focused crawlers (i.e., the ability to collect more relevant webpages), we used the harvest ratio metric. The harvest ratio is the percentage of crawled webpages that are relevant. The harvest ratio is similar to precision with regard to calculation (ratio of relevant to crawled/retrieved) but differs in the way of defining relevance. In precision, we compare to the true labels assigned manually. However, for harvest ratio, we use the labels assigned by the focused crawler as it proceeds, using either the event model or the baseline relevance estimation method.

6 Results

6.1 Event model-based versus topic-only classification

In this section, about the first series of experiments, we show the results of classifying the 1000 webpages about the California shooting using the topic-only vector space model versus three variants of our event model, namely topic + location, topic + date, and topic + location + date (our full event model).

Using the 38 seed webpages about the California shooting event, we created a vocabulary of 1365 keywords which appeared on 5 or more webpages. To extract the most representative keywords (features) from the vocabulary, we sorted the vocabulary keywords based on their cumulative normalized frequencies in all the seed webpages. We chose the top k keywords from the sorted vocabulary. Each seed webpage is then represented as a vector of the top k keywords along with their frequency of occurrence in the webpage. We created the topic reference vector as the centroid vector of all seed webpage vectors.

Table 1 The values of the parameters producing best F1-score

	URLs		Webpages	
	K	Threshold	*K*	Threshold
Topic-only	1310	0.25	10	0.45
Topic, Loc, and date	1310	0.15	10	0.4

K is the size of the topic vector, and threshold is the cutoff value for determining relevant or non-relevant labels based on the score

Table 2 Precision, recall, and F1-score for the four combinations of topic, location, date evaluated (URLs)

	Precision	Recall	F1-score
Topic	0.728	0.723	0.725
Topic + date	0.852	0.855	0.853
Topic + location	0.764	0.73	0.74
Topic + location + date	0.863	0.867	0.862

Table 3 Precision, recall, and F1-score for the combination of location and date with topic (webpages)

	Precision	Recall	F1-score
Topic	0.738	0.734	0.736
Topic + date	0.842	0.846	0.843
Topic + location	0.856	0.859	0.857
Topic + location + date	0.88	0.884	0.881

The 1000 URLs/webpages dataset about the California shooting consists of URL addresses and their anchor texts, as well as the corresponding webpages. These were tokenized, stopwords were removed, and words were stemmed. We ran separate experiments to classify URLs and webpages. The URLs and webpages were labeled manually as to relevant and non-relevant. Consequently, we refer to this dataset as labeled data.

As given in Table 1, for the topic-only approach, in the case of URLs, the values of the parameters k = number of keywords of topic vector and threshold that gave the best F1-score on the labeled data were 1310 and 0.25, respectively. In the case of webpages, they were 10 and 0.45, respectively. The weights for the topic, date, and location parts, calculated using Eqs. 1–4 described in Sect. 4.2.2, were 0.36, 0.22, and 0.42, respectively.

For the event model approach, in the case of URLs, the values of the parameters k = number of keywords of topic vector and threshold that gave the best F1-score on the labeled data were 1310 and 0.15, respectively. In the case of webpages, they were 10 and 0.4, respectively. The weights of topic, date, and location parts were 0.3, 0.355, and 0.345, respectively.

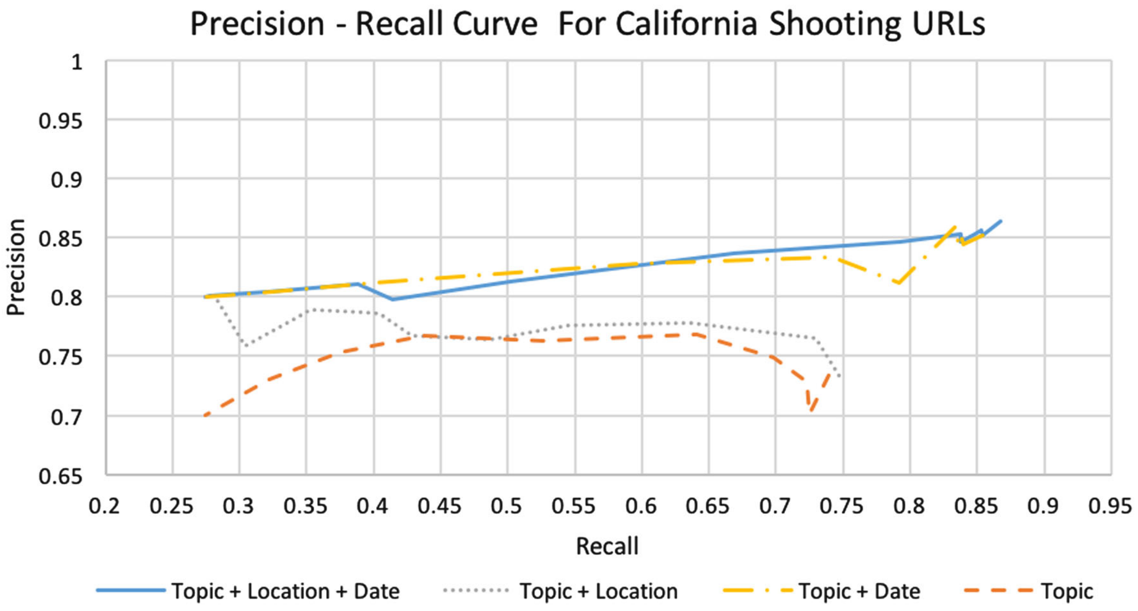


Fig. 7 Precision–recall curve for California shooting URLs evaluation

To examine the effect of the date and location separately, we ran our evaluation using topic + location and topic + date. For topic + location, the best threshold value was 0.2 and the weights of topic and location parts were 0.64 and 0.36, respectively. For topic + date, the best threshold value also was 0.2 and the weights of topic and location parts were 0.47 and 0.53, respectively.

Tables 2 and 3 show the precision, recall, and F1-score for the four experimental settings using the best values for the parameters for both the URL and webpage classification tasks (as shown in Table 1). Achieving higher F1-score means better classification performance (i.e., better ability to differentiate between relevant and non-relevant webpages).

The results show that adding date and/or location information to the topic enhances the performance. Our event model (combining topic, location, and date) achieves the best performance (highest F1-score). The topic-only model performed worst (lowest F1-score). Our examination of the data confirmed that it did not differentiate well between webpages talking about different shooting events and our event (California shooting), as all webpages are topically related to shooting. On the other hand, the topic + date model performed better than topic-only because it managed to use the publication date to filter out webpages talking about shooting events that happened before the California shooting. The topic + location model performed better than the topic-only model because it filtered out webpages talking about shooting events that happened at other locations than California.

We also examined the performance of our event model (combining topic, location, and date) versus the topic-only

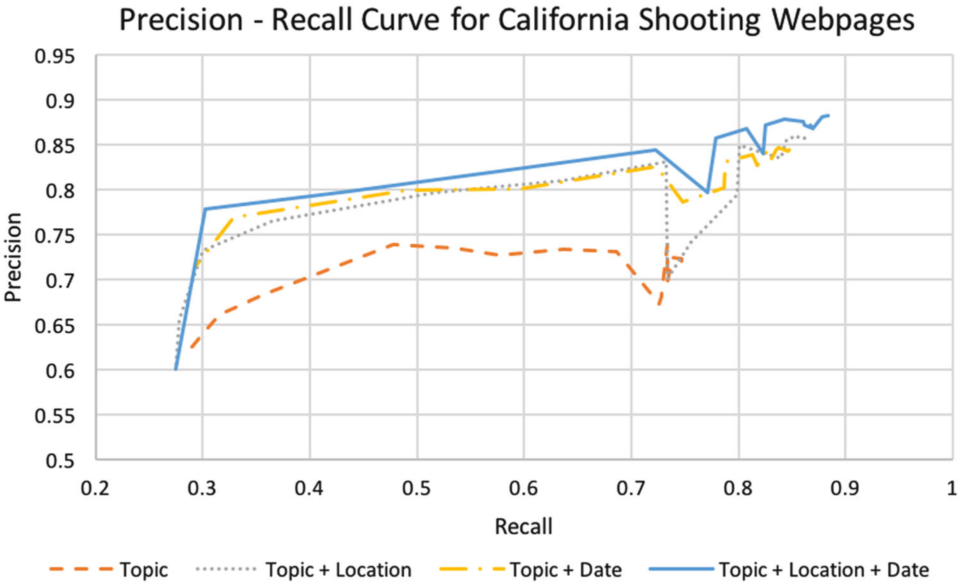


Fig. 8 Precision–recall curve for California shooting webpages evaluation

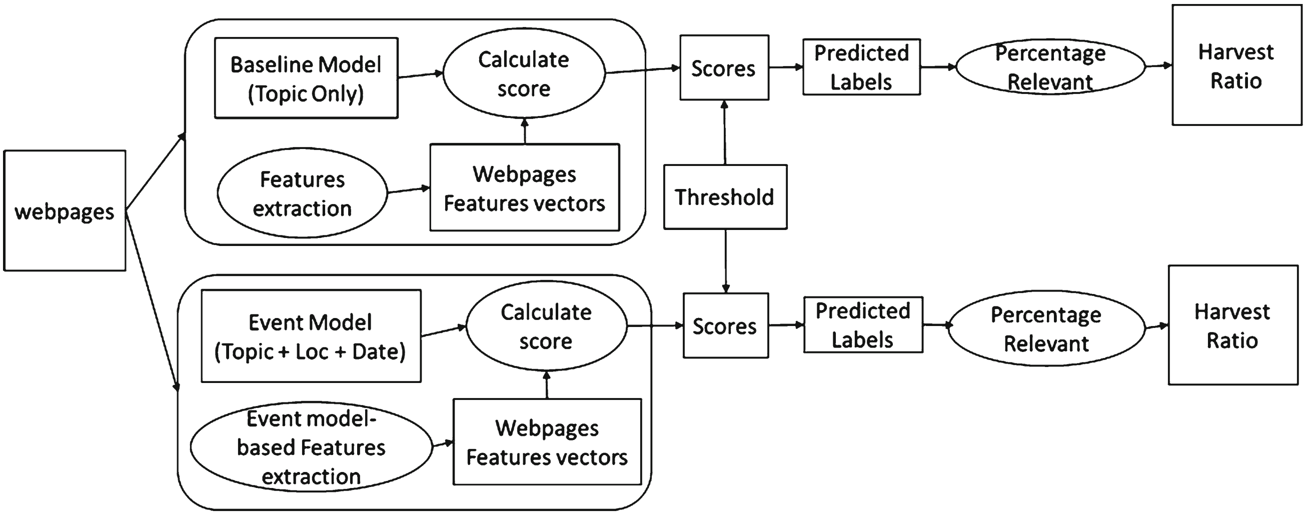


Fig. 9 The design of the experiment for evaluating the effectiveness of event model with focused crawler to retrieve relevant webpages

approach across the different values of the threshold variables and the best value for the *k* parameter (size of topic vector). We plotted the precision–recall curves for the different values of the threshold parameter. Figures 7 and 8 show the curves for the four different settings. The figures confirm the same result: Adding location and/or date information enhances the performance of classification. It is also shown that the effect of adding date information is much stronger than adding location information, in the case of URLs. We investigated this behavior and found that most of the URLs in our labeled data include date information that can be extracted easily. There was less location information in the URLs, as compared to date information. Further, some of the location information was not in a standard format as expected by SNER (which assumes location information exists as part of a sentence; see Sect. 4.2.3).

Table 4 California shooting event model

Keywords	Weights
Topic	
Shoot	0.93
San	0.513
Bernardino	0.465
Said	0.357
Wa	0.323
2015	0.321
Peopl	0.31
California	0.305
Polic	0.197
Suspect	0.177
Location	
San Bernardino	1.0
California	0.51
Calif.	0.44
Date	2015-12-02

6.2 Event model-based versus topic-only focused crawler

In this section, about the second series of experiments, we report the effect of using the event model with the focused crawler. Figure 9 shows the design and setup of the experiment for performing this evaluation. The first subsection is again about the California shooting. The second subsection is about the Brussels attack.

6.2.1 California shooting

In this experiment, we used the 38 URLs manually curated (see Sect. 5.1) as seeds for the two focused crawlers (our event model-based and the topic-only baseline). The event model built from the seeds is illustrated in Table 4. The first row in the table gives the topic vector keywords and their normalized cumulative term frequencies in all the seed webpages. Subsequent rows show the same for the location and date. We ran the two focused crawlers to collect 1000 webpages. We plot the percentage of crawled webpages that are relevant (harvest ratio) at different stages of the crawl, i.e., for the first 100, 200, 300, … crawled webpages. Figure 10 shows the performance of the two crawlers. Our event model-based focused crawler collected more relevant webpages during

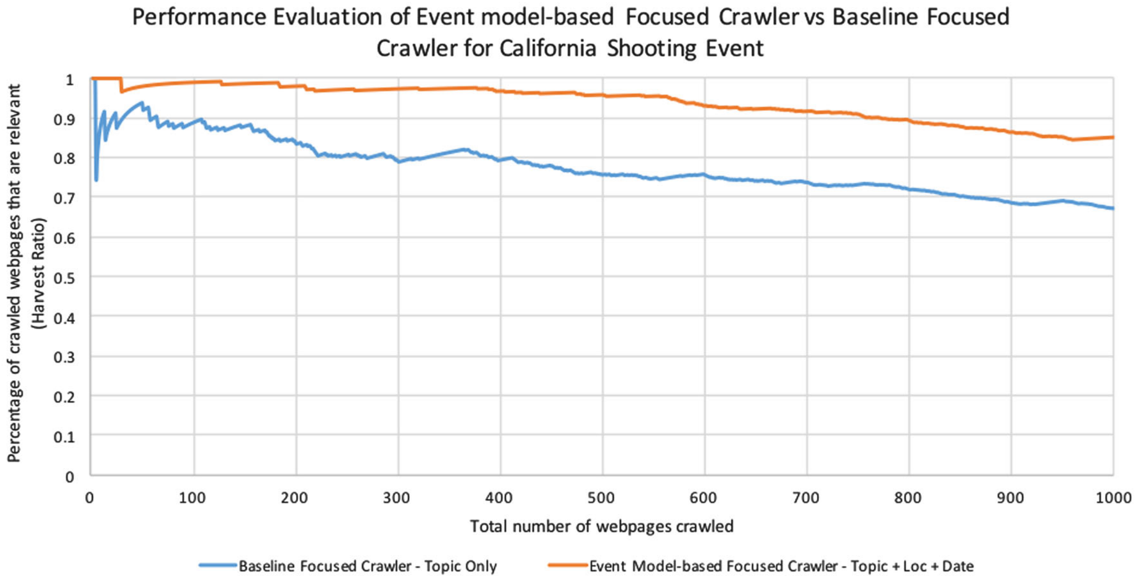


Fig. 10 Performance evaluation of event model-based versus topic-only focused crawlers for California shooting

the crawling process than the baseline topic-only focused crawler.

6.2.2 *Brussels attack*

In this experiment, about the Brussels attack, we used the 23 URLs manually curated (see Sect. 5.1) as seeds for the two focused crawlers (our event model-based and the topic-only baseline). The event model built from the seeds is illustrated in Table 5. The first row in the table gives the topic vector keywords and their normalized cumulative term frequencies in all the seed webpages. Subsequent rows show the same for the location and date. We ran the two focused crawlers to collect 1000 webpages. We plot the percentage of crawled webpages that are relevant (harvest ratio) at different stages of the crawl, i.e., for the first 100, 200, 300, … crawled webpages. Figure 11 shows the performance of the two crawlers. In like fashion to what is reported in Sect. 6.2.1, our event model-based focused crawler collected more relevant webpages during the crawling process than the topic-only baseline focused crawler.

Table 5 Brussels attack event model

Keywords	Weights
Topic	
Brussel	0.881
Attack	0.541
Airport	0.539
Explos	0.381
Wa	0.31
Peopl	0.273
Station	0.254
Belgium	0.242
Metro	0.197
Terror	0.159
Location	
Brussels	1.0
Belgium	0.37
Brussels airport	0.174
Zaventem	0.174
Paris	0.123
Date	2016-03-22

7 Conclusion and future work

We proposed a new model and representation for events. We showed how to represent an event using our model.

In a first series of experiments, we studied how well such a model can be used to classify webpages and URLs as to their relevance to an event. We calculated the weights of the three attributes of our event model by jointly optimizing two parameters—the number of keywords and the threshold value—to yield the best F1-score evaluation metric on a manually labeled (relevant and non-relevant) dataset of URLs and webpages about the California shooting. The results showed that the event model can effectively classify URLs and webpages to see whether they are relevant to the event of interest.

In a second series of experiments, we incorporated our event model into focused crawling and showed that our event model-based focused crawler can build an event-related Web collection more effectively than the state-of-the-art best-first topic-only focused crawler, on two different events: California shooting and Brussels attack. Our event model-based focused crawler outperformed the topic-only focused crawler by collecting more relevant webpages about the two events.

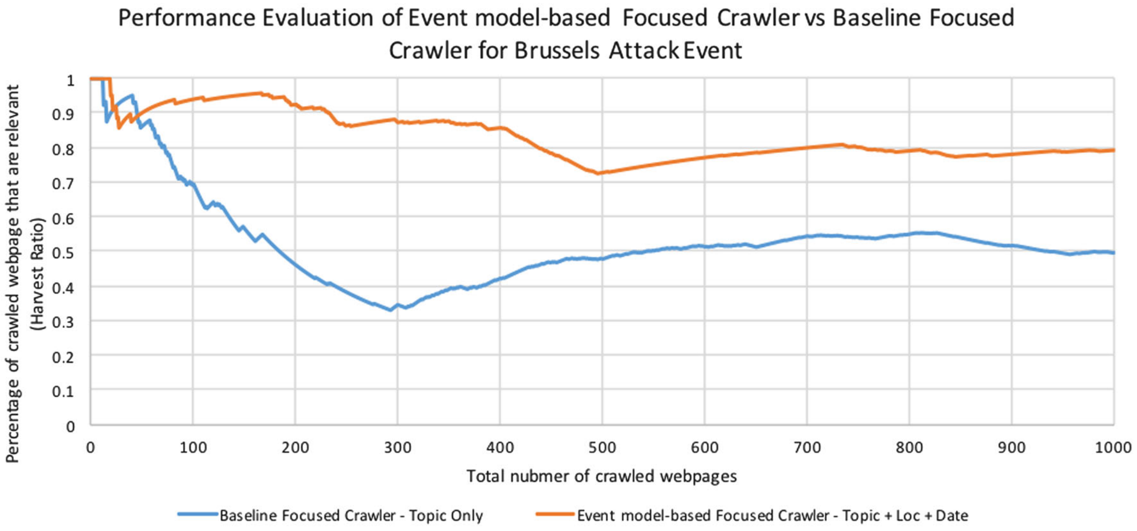


Fig. 11 Performance evaluation of event model-based focused crawler for Brussels attack

Our event model has captured three attributes for an event (topic, location, and date). We will extend our event model by extracting and adding organizations and participants; that information will represent the Who part in the Who did What, Where and When event model [31]. This will enrich our event model and consequently should increase the event model-based focused crawler's power to estimate relevance and retrieve more of the relevant webpages.

Further, with regard to focused crawling for large events, we are integrating our tweet collection efforts that already have resulted in over 1.3 billion tweets spread across about 1200 collection, with follow-up focused crawling that starts with seeds that come from the URLs found in those tweets. Preliminary results indicate our findings scale well with regard to building large sets of webpages for important events from seed sets found in tweets.

On the application side, we also plan to use our event model to analyze and summarize a collection of webpages; this can work for any collection about a particular event (e.g., prepared through manual curation, or using our event focused crawler). Using our event model, we could generate a list of indicative sentences and extract entities to represent and summarize an event. There are multiple algorithms and software implementations for text summarization, but we believe the concept of corpus/event summarization is new and worth investigation. Our preliminary investigation of such summarization suggests that results will have high quality and utility.

Acknowledgements Thanks go to NSF for support, especially through Grants IIS-1619028, CMMI-1638207, DUE-1141209, IIS-1319578, IIS-0916733, and IIS-0736055. Thanks also go to Virginia Tech's Digital Library Research Laboratory and Department of Computer Science. The statements made herein are solely the responsibility of the authors.

References

1. O'reilly, T.: What is web 2.0: design patterns and business models for the next generation of software. Commun. Strateg. **1**(1), 17 (2007)
2. Fox, E.A., Leidig, J.P.: Digital Libraries Applications: CBIR, Education, Social Networks, eScience/Simulation, and GIS, vol. 6. Morgan & Claypool Publishers, San Rafael (2014)
3. Fox, E.A., da Silva Torres, R.: Digital Library Technologies: Complex Objects, Annotation, Ontologies, Classification, Extraction, and Security, vol. 6. Morgan & Claypool Publishers, San Rafael (2014)
4. Shen, R., Goncalves, M.A., Fox, E.A.: Key Issues Regarding Digital Libraries: Evaluation and Integration, vol. 5. Morgan & Claypool Publishers, San Rafael (2013)
5. IDEAL. Integrated Digital Event Archive and Library. Accessed: 2016-07-26
6. Internet Archive. A digital library of free content and wayback machine. Accessed: 2016-07-26
7. Archive-It Collections. Spontaneous events. Accessed: 2016-07-26
8. Farag, M., Nakate, P., Fox, E.A.: Big data processing of school shooting archives. In: Proceedings of the 16th ACM/IEEE-CS on Joint Conference on Digital Libraries, pp. 271–272. ACM (2016)
9. IDEAL Collections. IDEAL event collections. Accessed: 2016-07-26
10. Archive-It. Web archiving services for libraries and archives. Accessed: 2016-07-26
11. Batsakis, S., Petrakis, E.G.M., Milios, E.: Improving the performance of focused web crawlers. Data Knowl. Eng. **68**(10), 1001–1013 (2009)
12. Chakrabarti, S., Van den Berg, M., Dom, B.: Focused crawling: a new approach to topic-specific web resource discovery. Comput. Netw. **31**(11), 1623–1640 (1999)
13. Pant, G., Srinivasan, P.: Learning to crawl: comparing classification schemes. ACM Trans. Inf. Syst. (TOIS) **23**(4), 430–462 (2005)
14. Rennie, J., McCallum, A.: Efficient web spidering with reinforcement learning. In: Proceedings of the International Conference on Machine Learning. Citeseer (1999)
15. Grigoriadis, A., Paliouras, G.: Focused crawling using temporal difference-learning. In: Hellenic Conference on Artificial Intelligence, pp. 142–153. Springer (2004)
16. Singh, N., Sandhawalia, H., Monet, N., Poirier, H., Coursimault, J.-M.: Large scale URL-based classification using online incremental learning. In: 2012 11th International Conference on Machine Learning and Applications (ICMLA), vol. **2**, pp. 402–409. IEEE (2012)
17. Menczer, F., Monge, A.E.: Scalable web search by adaptive online agents: an infospiders case study. In: Intelligent Information Agents, pp. 323–347. Springer (1999)
18. Dong, H., Hussain, F.K., Chang, E.: A survey in semantic web technologies-inspired focused crawlers. In: Third International Conference on Digital Information Management, 2008 (ICDIM 2008), pp. 934–936. IEEE (2008)
19. Ehrig, M., Maedche, A.: Ontology-focused crawling of web documents. In: Proceedings of the 2003 ACM symposium on Applied computing, pp. 1174–1178. ACM (2003)
20. Almpanidis, G., Kotropoulos, C., Pitas, I.: Combining text and link analysis for focused crawling—an application for vertical search engines. Inf. Syst. **32**(6), 886–908 (2007)
21. Diligenti, M., Coetzee, F., Lawrence, S., Giles, C.L., Gori, M. et al.: Focused crawling using context graphs. In: VLDB, pp. 527–534 (2000)
22. Pant, G., Srinivasan, P.: Link contexts in classifier-guided topical crawlers. IEEE Trans. Knowl. Data Eng. **18**(1), 107–122 (2006)
23. Kleinberg, J.M., Kumar, R., Raghavan, P., Rajagopalan, S., Tomkins, A.S.: The web as a graph: measurements, models, and methods. In: International Computing and Combinatorics Conference, pp. 1–17. Springer (1999)
24. Brin, S., Page, L.: Reprint of: The anatomy of a large-scale hypertextual web search engine. Comput. Netw. **56**(18), 3825–3833 (2012)
25. De Assis, Guilherme T., Laender, A.H.F., Gonçalves, M.A., Da Silva, A.S.: Exploiting genre in focused crawling. In: International Symposium on String Processing and Information Retrieval, pp. 62–73. Springer (2007)
26. Pant, G., Srinivasan, P.: Predicting web page status. Inf. Syst. Res. **21**(2), 345–364 (2010)
27. Pant, G., Srinivasan, P.: Status locality on the web: implications for building focused collections. Inf. Syst. Res. **24**(3), 802–821 (2013)
28. Chen, Y.: A novel hybrid focused crawling algorithm to build domain-specific collections. PhD thesis, Virginia Polytechnic Institute and State University (2007)
29. Allan, J.: Introduction to topic detection and tracking. In: Topic detection and tracking, pp. 1–16. Springer (2002)
30. Volkova, S., Caragea, D., Hsu, W.H., Bujuru, S.: Animal disease event recognition and classification. In: Proceedings of the First International Workshop on Web Science and Information Exchange in the Medical Web (MedEx 2010). Citeseer (2010)

31. Westermann, U., Jain, R.: Toward a common event model for multimedia applications. IEEE Multimed. **14**(1), 19–29 (2007)
32. Strötgen, J., Gertz, M., Junghans, C.: An event-centric model for multilingual document similarity. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 953–962. ACM (2011)
33. Farag, M.M.G., Fox, E.A.: Intelligent event focused crawling. In: Proceedings of the 11th International ISCRAM Conference. University Park, Pennsylvania, USA (2014)
34. Allan, J.: Topic Detection and Tracking: Event-Based Information Organization, vol. 12. Springer, Berlin (2012)
35. Gossen, G., Demidova, E., Risse, T.: iCrawl: improving the freshness of web collections by integrating social web and focused web crawling. In: Proceedings of the 15th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 75–84. ACM (2015)
36. AlNoamany, Y., Weigle, M.C., Nelson, M.L.: Detecting off-topic pages in web archives. In: International Conference on Theory and Practice of Digital Libraries, pp. 225–237. Springer (2015)
37. Salton, G., Wong, A., Yang, C.-S.: A vector space model for automatic indexing. Commun. ACM **18**(11), 613–620 (1975)
38. Menczer, F., Pant, G., Srinivasan, P.: Topical web crawlers: evaluating adaptive algorithms. ACM Trans. Internet Technol. (TOIT) **4**(4), 378–419 (2004)
39. Klein, M., Shipman, J., Nelson, M.L.: Is this a good title? In: Proceedings of the 21st ACM Conference on Hypertext and Hypermedia, pp. 3–12. ACM (2010)
40. Finkel, J.R., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by gibbs sampling. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pp. 363–370. Association for Computational Linguistics (2005)
41. Baeza-Yates, R., Ribeiro-Neto, B., et al.: Modern Information Retrieval, vol. 463. ACM press, New York (1999)