Event Detection and Tracking in Social Streams [*]

Hassan Sayyadi
University of Maryland-College Park,Microsoft Research & LiveLabs, sayyadi@cs.umd.edu

Matthew Hurst and **Alexey Maykov**
Microsoft LiveLabs, {mhurst,amaykov}@microsoft.com

Abstract

Events and stories can be characterized by a set of descriptive, collocated keywords. Intuitively, documents describing the same event will contain similar sets of keywords, and the graph of keywords for a document collection will contain clusters individual events. In this paper we build a network of keywords based on their co-occurrence in documents. We propose and develop a new event detection algorithm which creates a keyword graph and uses community detection methods analogous to those used for social network analysis to discover and describe events. Constellations of keywords describing an event may be used to find related articles. We also use the proposed algorithm to analyze events and track stories in social streams.

Introduction

We can define a news event as being any event (something happening at a specific time and place) of interest to the (news) media. Furthermore, we can consider any such event as being a single episode in a larger story arc. For example, a speech at a rally might be an event, but it is an episode in a larger context: a presidential election. To help position our work, we will use the term *episode* to mean any such event and *saga* to refer to the collection of events related within a broader context. The key challenges that the work in this paper addresses are: the detection of episodes, and the formation of sagas.

If we think about episodes abstractly, there is a process of selection and ranking which results in their being reported in media. Social Media (which we define by extension as blogs, usenet, message boards, etc.) can be considered evidence of an example of that type of process. The attention that we discover in this content to main stream news articles, or just to events reported in isolation, indicates what is important to the authors. Taken in aggregate, we can analyze this attention to discover important (or news-worthy) events. In addition, by analyzing social media in this way, we can compare it with a different selection and ranking process: that of the main stream media, their reporters and editors.

Related Work

New Event Detection (NED) models usually do a single pass incremental clustering algorithm. For a newly arrived document the similarity between the document and known events is computed and the maximum similarity will be selected. If the similarity is more than the predefined threshold, the document will be assigned to the corresponding event, otherwise it will be considered a new event.

Allen et al. (Allan, Papka, and Lavrenko 1998) used a modified version of *TF/IDF* and also penalized the threshold by the time distance between the document and the event. Since future document features are not known, such online clustering algorithms need to estimate *IDF*. While Allen et al. (Allan, Papka, and Lavrenko 1998) use an auxiliary dataset to estimate *IDF*, Yang et al. (Yang, Pierce, and Carbonell 1998) propose an *incremental IDF* factor. Yang et al. consider a time window and also a decay factor for the similarity between documents and events based on the time difference.

Yang et. al (Yang, Pierce, and Carbonell 1998) proposed an agglomerative clustering, GAC (augmented Group Average Clustering), to extract retrospective events in the corpus. They also applied an iterative bucketing and re-clustering model proposed by Cutting et al. (buc) to compromise between cluster quality and computational efficiency. Li et al. (Li et al. 2005) proposed a probabilistic model for RED. and use Expectation Maximization (EM) algorithm to maximize log-likelihood of the distributions and learn the model parameters. Such algorithm requires the number of events to be given which is difficult in practice. Li et al. computed an estimation of event counts from the article count-time distribution. While most of event detection models use similar algorithms, many variations of the document representation, distance or similarity metrics, and clustering algorithm are proposed in the literature (Kumaran and Allan 2004; Brants, Chen, and Farahat 2003; Yang et al. 2002; Lam et al. 2001). Mory et al. (Mori, Miura, and Shioya 2004; 2006) used features extracted from the KeyGraph (Ohsawa, Benson, and Yachida 1998) in which each maximally connected component, which is called a foundation or basic concept, is chosen as a document feature. Documents repre-

[*]This research was partially supported by National Science Foundation grant CMMI 0753124 INTEROP: Rapid Deployment of Humanitarian Assistance Social Networks for ad hoc Geospatial Data Sharing (GeoNets)

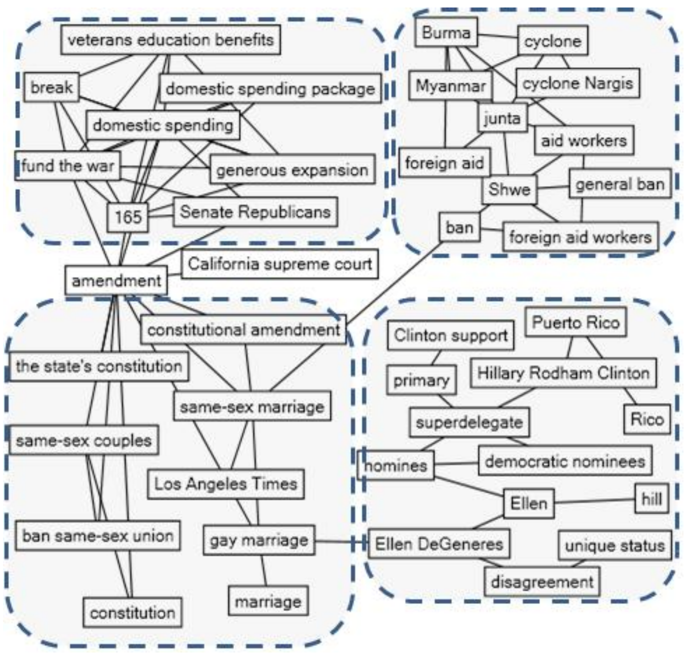

Figure 1: A sample of the KeyGraph and communities of keywords

sented with basic concepts, were clustered by a *single-link* clustering algorithm. Toda et al. (Toda and Kataoka 2005) and Sayyadi et al. (Sayyadi, Salehi, and Abolhassani 2006) have also proposed *Label-Based clustering* algorithms for search result clustering of news engines to overcome the performance issue of ordinary *Cluster-based clustering* models.

Proposed Algorithm

Both news and events can be represented by keywords and there are several ways in which keywords can be extracted from articles. We might compute a set of terms that maximally distinguish a document from some background set of documents. Alternatively, we might extract the set of named entities, or even noun phrases found in the document. Key to this approach is the interdependency between extracted terms. A document summarized by the terms 'Hillary' and 'Clinton' could not be used to discover related documents that only mention 'Hillary' or only mention 'Clinton'. In addition, a document summarized by a set of terms does not imply that another episode may not be described by some subset of those terms. We will see later how our proposed algorithm deals with this episode-sense ambiguity at the keyword level. Our algorithm uses a graph of extracted terms - a KeyGraph (Ohsawa, Benson, and Yachida 1998; Mori, Miura, and Shioya 2006; 2004). Nodes are the keywords and edges between the nodes are formed when those terms co-occur in a document. We apply community analysis techniques to this graph adapted from social network theory to discover events (communities of nodes)(Figure 1). This figure is a small part of the keyword graph obtained from posts published in May 22nd and 23rd. As it is clear in the figure, there are several communities of keywords which are suggestive of specific events. Inside each community, nodes are highly connected and the connection to the nodes outside of the community is very rare. For example, the group of keywords including "*cyclone*", "*Myanmar*", "*foreign aid workers*", etc create a community for the *the cyclone in Myanmar* and the Myanmar government did not accept international aid in the first days.

Building the KeyGraph A keygraph is built by first extracting a set of keywords. Then, for each keyword k_i we calculate the term frequency ($TF_{i,j}$), document frequency (DF_i) and the inverse document frequency (IDF_i). Keywords with low document frequency are filtered, and a node (n_i) in the KeyGraph is created for each remaining keyword (k_i). Then, an edge $e_{i,j}$ between nodes n_i and n_j is be added if k_i and k_j co-occur in the same document. To reduce the noise in the data, each edge should satisfy two conditions: An edge is removed if the keywords associated with its nodes co-occur below some minimum threshold. The second condition relates to the conditional probability of the edge. For $e_{i,j}$, the conditional probability of the occurrence $p(k_i|k_j)$ (the probability of seeing k_i in a documents if k_j exists in the document), and similarly $p(k_j|k_i)$ are calculated and if both of them are smaller than the defined threshold, the edge will be removed. By these conditions the correlation between nodes will be tested and noisy edges between independent nodes will be removed.

Community Detection in KeyGraph We make the assumption that keywords co-occur when there is some meaningful topical relationship between them. By analogy, we can think of the graph as a social network of relationships between keywords. As Figure 1 makes clear, communities of keywords (those sets of keywords which together capture some meaningful topic via the relationships between the members) are densely linked, while there are few links between nodes from different topical communities. The betweenness centrality score is a good measure to find the edges between two communities. Betweenness centrality for a node is defined as the number of shortest paths between all pairs of nodes that pass through that node. Inter-community edges will always obtain a high score, since the shortest paths between nodes from different communities will pass through them. After removing the edges with a high betweenness centrality score, every connected component of the KeyGraph represents a hypothesis about an event, the keywords forming a bag of words summary of the event. The algorithm removes edges iteratively. The set of shortest paths between all pairs of nodes is found by breadth first search. Then, the edge belonging to the most paths is selected for removal from the graph. If two edges have the same betweenness centrality score, the one with the lower conditional probability will be removed.

Of course, a keyword can appear in more than one event. However, the original approach to community detection by using betweenness centrality score of edges does not support it. In the basic model, since only edges are removed, each node cannot appear in two communities. We extend the original algorithm to make it more flexible and allow nodes

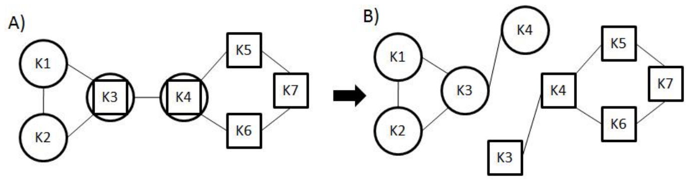

Figure 2: Edge and node duplication to allow nodes be in more than one cluster

to be duplicated. Before removing an edge, we first check a condition. If the edge's conditional probability is high, the edge and corresponding nodes will be duplicated (Figure 2).

After removing or duplicating one edge in each iteration, the betweenness centrality score in the next iteration will be computed again for the new graph and the same steps will be repeated until all edges with the high score are removed. Finally, the communities of keywords will be used to find the communities of documents which may represent a collection of documents for events.

Document Clustering Each community of keywords may be thought of as a synthetic document, called a *key document*. Documents in the original corpus which are similar to this synthetic document can be clustered, thus retrieving a cluster of topical documents. We use cosine similarity to discover document clusters for key documents.

In some cases, the keywords in a key document are general or the key document contains only a few keywords. This results in a general category of documents. Usually, such key documents create a cluster of documents belonging to a high level category or a long story. For example, a key document contains only "Obama", "Clinton", "vote", and "election" points to the USA 2008 election which is not a discrete event. Hence, the similarity of the documents belonging to a key document help to find such key documents, since the documents will discuss a broad number of smaller topics and they will be different. Considering documents as a data point, the variance of documents belonging to such key documents will be very high. Consequently, the next step would be to calculate the variance of documents for each key document and then filter key documents with high variance. This helps to find key documents that truly represent events. Although documents are assigned to key clusters based on the similarity of documents to the key documents, some of documents may be filtered later by other conditions to reduce noise. We also find the similarity between the documents and the centroid of each cluster and filter documents with low cosine similarity to the centroid. Furthermore, an article can be assigned to more than one event. We find the document overlap of events and then merge those with the high document overlap.

Experimental Result

The data used in our experiments was drawn from Live Labs's Social Streams platform. The dataset used in the following experiments is a set of blog posts from May and June 2009. These posts were filtered to include only those with at least one link to a news article. We used a knowledge driven url classifier to determine if a link was to a news article or not. The final dataset contains 18,000 posts for two months, an average of 3,000 posts per day.

Keyword Extraction and Keyword Community Detection We used three approaches to extract keywords for each document: (1) the algorithm proposed by (Dunning 1993) for keyword extraction (2) extracting noun phrases as keywords from each document and (3) extracting named entities (person, place, organizations and monetary amounts) as well as noun phrases. For all of three approaches all stopwords are removed and all extracted keywords are stemmed by a porter stemmer. Generally, named entities appear as a noun phrase in the text but in many cases they are used as a part of noun phrase in combination with other words, so we add them as individual words separately. The number of extracted events per day using the three approaches shows that extracting noun phrases and named entities as keywords for documents outperforms the other approaches. Instead of calculating the betweenness centrality score for large ketyraphs, we computed the approximated betweenness centrality of edges. The approximated score is obtained by finding the shortest paths between pairs of nodes which are sampled randomly from all possible pairs of nodes. To evaluate the accuracy of the approximated scores we select many large graphs and run the community detection algorithm with both exact and approximated scores, then compare the communities. The experiments showed that the extracted communities are exactly the same communities for both approaches. In fact, we only use the approximated score for large graphs and, and when components are broken to smaller components by eliminating edges with the high approximated score, we compute the exact scores for the small components and eliminate edges with high exact scores.

Temporal Analysis Intuitively, events have a temporal characteristic, so events extracted by the event detection algorithms should show some sort of temporal cohesion. Basically, news events will be reported by many news articles in the period immediately following, and over the time the number of articles reporting an event will decrease until nobody writes about that event. We also run the event detection algorithm on the blog posts collected during June 2008 then calculate the time span of detected events. Figure 3 shows the cumulative percentage of events based on the average publication time difference of articles. Similar to the average distances, the standard deviations of 90% of events are less than 11 days. As shown in Figure 4 the number of articles per day decrease for each event which shows the results of our algorithm has the true temporal characteristic of news articles.

Story Tracking in Social Streams Social Media content is produced at a high volume, potentially hundreds or even thousands of posts per second. Hence, it is practically impossible to repeat the process of event detection on the entire data when a new article is adding to the dataset. We keep a

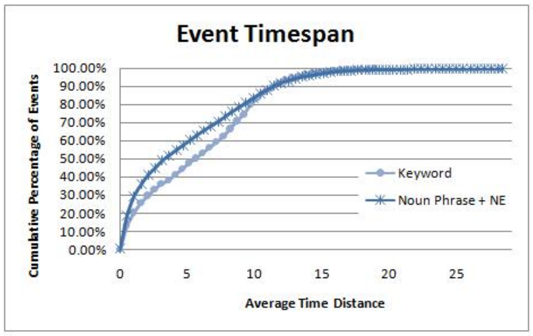

Figure 3: Cumulative percentage of events based on the average publish time difference of articles

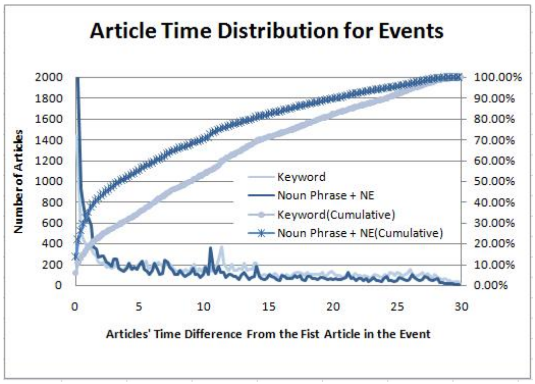

Figure 4: Articles distribution based on the distance from the fist article for events

sliding window of articles and find all events for the articles in the current window. Then related events can be found by the document overlap of events from the current window and events in the previous window.

Conclusion and Future Work

In this work we propose a new algorithm for event detection using the co-occurrence of keywords. In our community detection algorithm, nodes can fall into different communities as a word or phrase can be in keywords list of more than one event. In the current version of our algorithm we count all keywords in one community as keywords for the event, though a subset of keywords may be better, especially in cases where the number of nodes is large. In addition, while the keyword graph is a weighted graph, in order to find the betweenness centrality score, shortest paths are found on an un-weighted graph. Hence, another future direction of research is to take into account the weight of edges in finding shortest paths.

References

Allan, J.; Papka, R.; and Lavrenko, V. 1998. On-line new event detection and tracking. In *SIGIR'98: Proceedings of the 21st Annual International ACM SIGIR conference on Research and development in information retrieval.*

Brants, T.; Chen, F.; and Farahat, A. 2003. A system for new event detection. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, 330–337. New York, NY, USA: ACM.

scatter/gather.

Dunning, T. 1993. Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.* 19(1):61–74.

Kumaran, G., and Allan, J. 2004. Text classification and named entities for new event detection. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, 297–304. New York, NY, USA: ACM.

Lam, W.; Meng, H. M. L.; Wong, K. L.; and Yen, J. C. H. 2001. Using contextual analysis for news event detection. *Int. J. Intell. Syst.* 16(4):525–546.

Li, Z.; Wang, B.; Li, M.; and Ma, W.-Y. 2005. A probabilistic model for retrospective news event detection. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, 106–113. New York, NY, USA: ACM.

Mori, M.; Miura, T.; and Shioya, I. 2004. Extracting events from web pages. In *AISTA'04: Proceedings of the International Conference on Advances in Intelligent Systems - Theory and Applications (AISTA).*

Mori, M.; Miura, T.; and Shioya, I. 2006. Topic detection and tracking for news web pages. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, 338–342.

Ohsawa, Y.; Benson, N. E.; and Yachida, M. 1998. Keygraph: Automatic indexing by co-occurrence graph based on building construction metaphor. In *ADL '98: Proceedings of the Advances in Digital Libraries Conference*, 12.

Sayyadi, H.; Salehi, S.; and Abolhassani, H. 2006. Nesrec: News meta-search result clustering". In *n Proceeding of CIS2E 06 The International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering*, 173–178.

Toda, H., and Kataoka, R. 2005. A search result clustering method using informatively named entities. In *WIDM '05: Proceedings of the 7th annual ACM international workshop on Web information and data management*, 81–86. New York, NY, USA: ACM.

Yang, Y.; Zhang, J.; Carbonell, J.; and Jin, C. 2002. Topic-conditioned novelty detection. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 688–693. New York, NY, USA: ACM.

Yang, Y.; Pierce, T.; and Carbonell, J. G. 1998. A study on retrospective and on-line event detection. In *SIGIR'98: Proceedings of the 21st Annual International ACM SIGIR conference on Research and development in information retrieval.*