From Episodes to Sagas: Understanding the News by Identifying Temporally Related Story Sequences

Ramnath Balasubramanyan* **Frank Lin***
William W. Cohen* **Matthew Hurst†** **Noah A. Smith***
*Language Technologies Institute, Carnegie Mellon University
†Microsoft Corporation

Abstract

Current news interfaces are largely driven by recent information, even though many events are better interpreted in context of previous related events. To address this problem, we consider the task of constructing an explicit representation of a "saga"—i.e., a long-running series of related events. We define a timeline as a concrete representation of a "saga" and we propose two unsupervised methods for timeline construction and compare their performance to manually-produced timelines using a tree edit distance-based measure. Preliminary results using these techniques on a weblog corpus and a supplementary news corpus are presented and show both promise and challenges.

Introduction: Why Timelines Are Useful

According to recent surveys, the Internet is rapidly replacing print as the primary news source for many people. However, the large quantity of available news sources on the Internet poses new interface challenges.

One limitation of most current news interfaces is that they are largely driven by recent information: most of the user's attention is directed toward events of the last few hours or even minutes. This leads to a view of current events which is broad, but shallow. Many events are better interpreted in context of previous related events. For example, Figure 1 shows an summary of such an event, circa Jan 15, 2009, involving statements made by Eric Holder, the nominee for Attorney General. This event is best interpreted in the context of a long-running controversy over interrogation techniques and human rights—a controversy which may not be immediately obvious to a reader who has not been actively tracking these events. Figure 2 shows a small section of a "timeline" of events from this long-running event series.

The hypothesis behind our work is that it is useful to construct explicit representations of such "sagas"—i.e., long-running sequences of related events. Indeed, summaries of these sagas have been manually produced by various authors for many such event sequences (e.g., Figure 2). From an application perspective, we are interested in providing tools that will give a reader the complete narrative context of any given event. From a sociological perspective, we are interested in finding out how real world events are perceived, reported and synthesized in a number of media types including news and weblogs.

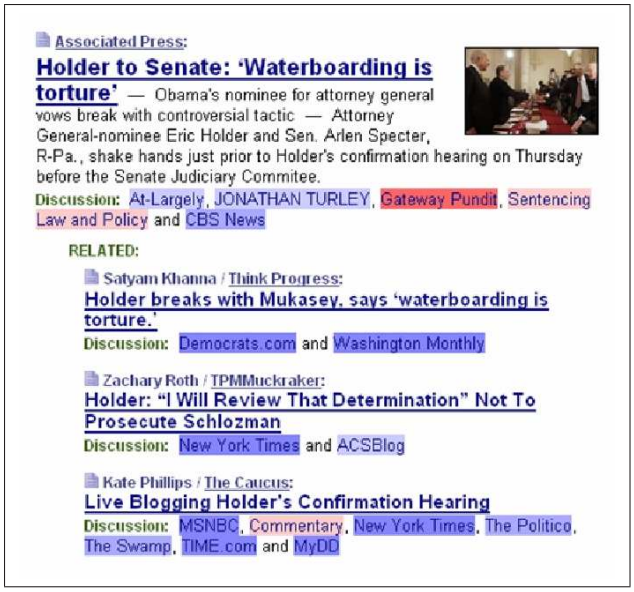

Figure 1: Two clusters of events, from Memeorandum.com screenshots.

To more precisely ground the problem we will propose a simple model of events and their relationships. An event, represented by a node in an *event graph*, has a time and duration, and is described in some appropriate manner (e.g., a textual description, a logical expression, set of normalized entities, etc.). The edges in an event graph encode binary relations between events. These relations could be simple temporal relationships (e.g., "event a precedes event b", "event a temporal overlaps with event b") but could also capture causality or other deeper relationships.

We think of textual data as being generated by (or more broadly, associated with) an underlying latent event graph. This viewpoint leads immediately to a number of technical challenges involving completing incompletely-specified event graphs such as: discovering latent event graphs from an observed corpus, adding new events to a partial event graph, and constructing mappings between documents (or

- ...
- 9 August – The Pentagon announces that the CSRTs had determined that all 14 detainees transferred to Guantnamo in September 2006 met the criteria for designation as "enemy combatants."
- 6 December - The CIA Director reveals that videotapes of interrogations conducted in 2002 held in the CIA's secret detention program had been destroyed by the agency in 2005. The tapes may have included a record of the use of the torture method known as "waterboarding" – simulated drowning – and other so-called "enhanced" interrogation techniques used by the CIA.

2008

- 5 February - The CIA Director confirms that "waterboarding" was used in 2002 and 2003 by the agency as an interrogation technique against three detainees held in secret custody.
- 14 March - The Pentagon announces that it has transferred Afghan national Muhammad Rahim al-Afghani to Guantnamo. Prior to his transfer he had been held in secret CIA custody.
- ...

Figure 2: Part of a timeline manually created by Amnesty International [www.amnestyusa.org].

parts of documents) and events in a graph. [1] These technical challenges are discussed below in more detail.

While the general notion of an event graph is useful, in this paper we focus on two special cases. One special case is a simple *timeline*—i.e., a linear sequence of events (as shown in Figure 2). We will also consider graphs in which events are partially ordered by *inclusion*—i.e., where some longer "abstract" events a completely contain shorter "concrete" subevents b. As an example, an event like "the 2008 election" might include the subevents "the Democratic primary" and "the Republican primary."

In this work we will mine timelines from weblogs. One potential advantage using social media is that it provides information about the relative importance of events, as perceived by members of a social community, thus provided a more normative view of what events should be in a timeline. Likewise, social media provides information about the appropriate granularity of events. (Theoretically, any event can be broken down into subevents, and so on; the level of event granularity most appropriate to a particular community is arguably best discovered from analysis of how that community discusses events.)

Challenges in Constructing Timelines

Definitions

Our long-term goal is to automatically construct a cohesive narrative for a "saga" that is easily accessible to the user and that facilitates in-depth study of news on any topic. This is a difficult task, because understanding which past stories give the best context for an event is difficult, requiring many subtle judgments about relevance, entity identity, and so on. There are also a number of less immediately obvious challenges that we will discuss below; we will begin, however, by proposing a precise notion of a *timeline*.

In this paper, a *timeline* TL is a sequence of *event nodes* $n_1, \ldots, n_k$, each of which corresponds to an *event* e_i, by which we mean, informally, something that happened in the "real world." Each event node n_i has an associated *time span* t_i, indicating the duration of the associated event, and a *textual description* q_i (e.g., "Holder confirmed as Attorney General").

Given a particular corpus C of documents—e.g., a collection of blog postings or news stories—an event node n_i can also be associated with a binary classifier r_i^C, which labels each document in $d \in C$ with an indicator as to whether or not it is relevant to event e_i. We will call r_i^C an *event classifier*. A common way of summarizing a timeline on the web is to provide, for each event node n_i, the time span, a description, and a small sample of relevant documents (maybe only one or two).

In summary, then, for this paper we will define a *complete timeline* $T(C)$ over a corpus C as a set of *event nodes* $n_1, \ldots, n_k$, each which has the following properties:

- a short textual description q_i;
- an associated real-world event e_i;
- a *time span* t_i indicating the duration of e_i, where t_i is further defined by a start and end time;
- an indication of which documents $d \in C$ are relevant to n_i, represented as a function $r_i^C(d)$, where $r_i(d) = 1$ iff d is relevant to n_i;
- a sample S_i^C of highly-relevant documents from C.

Timeline Completion Tasks

Notice that e_i is different in character from the other properties of n_i—it is purely conceptual (the real-world referent of n_i), and will not be explicitly provided by the user. Other parts of $T(C)$ will also typically be missing. For instance, a timeline author might specify each description q_i and provide a sample of relevant documents $S_i^C = s_i^1, \ldots, s_i^{\ell_i}$, but an author is unlikely to provide a precise duration for the event e_i. Authors are even less likely to provide complete relevance judgments for all documents in C for each event e_i. Authors might also be aware of only some of the events in a timeline.

We will use the term *timeline completion* for completing a partially-specified timeline. Each kind of incompletely-specified timeline leads to a slightly different technical different problem, many of which can be mapped to well-studied tasks in learning, natural language processing, and information retrieval; for example (in each case we assume C is given):

- If a text description q_i is given, finding a small sample of relevant documents S_i^C is an information retrieval problem. One challenge that differs from traditional information retrieval tasks is finding a few documents that with

[1] In this paper, we assume that the text describes objectively true events, thus ignoring issues regarding the fidelity of the text, the bias of the author, and so on.

succinct and objective description of the event without too much information overlap with sample documents of other events in the same timeline.

- If each sample S_i^C is given, then finding the event classifiers r_i^C is a semi-supervised classification problem—if one makes the additional assumption that the documents relevant to each event e_i are disjoint (or not disjoint, which makes it a more complicated and challenging classification problem).
- If either S_i^C or r_i^C is given, finding a good textual description q_i is a summarization problem.
- If r_i^C is given but S_i^C is not, then finding S_i^C is equivalent to finding representative exemplars of a class.
- If only C is given, then finding r_i^C is an unsupervised clustering problem—ideally one that should be performed using the dynamic nature of the corpus (Yang, Pierce, & Carbonell 1998; Wang & McCallum 2006; Blei & Lafferty 2006).

Additionally, there are many other timeline-related tasks that do not neatly correspond to well-studied technical tasks: for instance, finding in-depth articles that discuss many events in a timeline, or comparing two corpora using the "lens" of a timeline (e.g., to identify differences in the types of events discussed by the two corpora).

In the remainder of this paper, we will focus on the most ambitious of these tasks—unsupervised construction of a timeline from a corpus. As a simplification, however, we will ignore the problem of summarization. We will also explore a slight relaxation of this task in which some information about the duration of events is provided.

We first describe the corpora that we will use in our experiments, and then discuss two unsupervised approaches, one based on probabilistic modeling, and one based on network analysis. We then consider manually-generated timelines, and discuss the types disagreement and level of disagreement between annotators. We also present quantitative results measuring the agreement between annotators and the unsupervised methods before concluding.

Corpora

The primary data we used in the experiments is a collection of roughly 5.5 million blog posts published during the month of May 2008. The blog posts are gathered from a number of blog services (such as TypePad, Vox, and LiveJournal) owned by the blogging service company Six Apart. In addition, we supplemented the blog data with about 27 thousand news articles obtained by following links found in the blog posts.

The weblog and news data was gathered using Microsoft Live Labs' Social Streams platform, which gathers social media in real time using a number of special purpose crawlers. News articles are identified in the stream of social content and collected in a parallel collection system.

The raw data is in atom stream format; we parse the atom stream and from each blog post we extract the blog title, post title, publish date, post content, and links within the post content. The post content, which often contain various HTML tags and formatting symbols and characters are cleaned up as much as possible to result in a plain text content. All the fields are then stored in an Apache Lucene database and indexed based on the text tokens in the post title and post content for full-text search.

For experimental results, we focus on a subset of the blog corpus that is related to the US Presidential Democratic primaries. The top 1000 or 2000 documents returned by issuing the query "hillary obama" to the Lucene blog post index were used. In addition, the news corpus comprising of the news stories linked to from the blogs is also used in the probabilistic modeling approach.

Finding Timelines with Generative Models

A commonly used generative model for unsupervised clustering of text is the mixture of multinomials. In this model, each topic is represented by a multinomial distribution over words. Each document is generated by such a multinomial conditioned on the cluster it belongs to. An extension to this model is the *SpeClustering* model, proposed by Huang and Mitchell (Huang 2006). In this model, words in a document are generated by either a topic specific distribution or a general word distribution that captures the non topic specific content in the document.

Here we extend the SpeClustering model by additionally modeling the timestamps of the documents in the corpus. Associated with each topic is a Gaussian distribution that generates the timestamp of the document. The model is shown graphically in Figure 3.

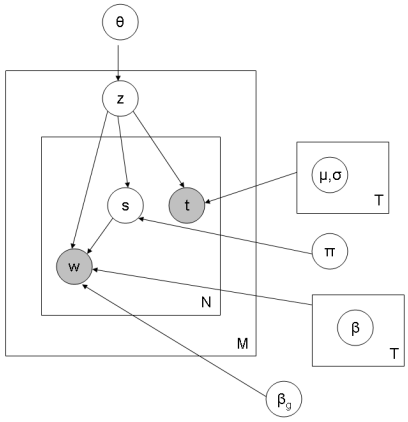

Figure 3: The SpeCluster over Time model

For a corpus C with M documents, N_i words in each document, ρ_i being the timestamp $\forall i$=1, . . . , M and T topics, the model has the following parameters: θ: multinomial over topics, β: per-topic distribution over words, β_g: general word distribution, π: topic-specific binomial indicating the proclivity towards using the topic specific distribution, $[\mu, \sigma]$: per-topic Gaussian parameters for timestamp distributions, z: topic variable of a document, s: boolean variable which indicates if the word was generated from the general

word distribution or the topic specific distribution, t: observed timestamp which indicates the time of the event reported in the document, and w: observed word in the document.

Inference

A maximum likelihood estimate of the parameters that maximizes the likelihood of observing the corpus can be obtained by running an EM procedure. The following quantities are computed in the E-step:

$$\begin{aligned}\phi_i^t(z) &= \frac{\theta_z^t \Pi_{j=1}^{N_i} \left[\pi_z^t \beta_{zo_{ij}}^t + (1-\pi_z^t)\beta_{go_{ij}}^t\right]}{\sum_{k=1}^{T} \theta_k^t \Pi_{j=1}^{N_i} \left[\pi_z^t \beta_{ko_{ij}}^t + (1-\pi_z^t)\beta_{go_{ij}}^t\right]} \\ \psi_{ij}^t(z) &= \frac{\pi_z^t \beta_{zo_{ij}}^t}{\pi_z^t \beta_{zo_{ij}}^t + (1-\pi_z^t)\beta_{go_{ij}}^t}\end{aligned}$$

The exact estimation for each parameter in the M step is as below:

$$\begin{aligned}\theta_z^{t+1} &= \frac{\sum_{i=1}^{M} \phi_i^t(z)}{M} \\ \pi_z^{t+1} &= \frac{\sum_{i=1}^{M} \phi_i^t(z) \sum_{j=1}^{N_i} \psi_{ij}^t(z)}{\sum_{i=1}^{M} \phi_i^t(z) N_i} \\ \beta_{zv}^{t+1} &= \frac{\sum_{i=1}^{M} \phi_i^t(z) \sum_{j=1}^{N_i} \delta(w_{ij}=v)\psi_{ij}^t(z)}{\sum_{i=1}^{M} \phi_i^t(z) \sum_{j=1}^{N_i} \psi_{ij}^t(z)} \\ \beta_{gv}^{t+1} &= \frac{\sum_{i=1}^{M} \sum_{k=1}^{T} \phi_i^t(k) \sum_{j=1}^{N_i} \delta(w_{ij}=v)\psi_{ij}^t(k)}{\sum_{i=1}^{M} \sum_{k=1}^{T} \phi_i^t(k) \sum_{j=1}^{N_i} \psi_{ij}^t(k)} \\ \mu_z^{t+1} &= \frac{\sum_{i=1}^{M} \phi_i^t(z)\rho_i}{M} \\ \sigma_z^{2t+1} &= \frac{\zeta^2 + \sum_{i=1}^{M} \phi_i^t(z)\left(\rho_i - \mu_z^{t+1}\right)^2}{\nu + 2 + \sum_{i=1}^{M} \phi_i^t(z)}\end{aligned} \quad (1)$$

In equation 1, an inverse gamma prior is placed on the variances of the class specific normal distributions. The pdf of an inverse gamma distribution is given by:

$$p\left(\sigma^2 | \frac{\nu}{2}, \frac{\zeta^2}{2}\right) \propto \sigma^{2\left(\frac{\nu+2}{2}\right)} e^{\left(\frac{-\zeta^2}{2\sigma^2}\right)}$$

Derivation of the update rule is presented in detail in (Fraley & Raftery 2007).

Relationship to Other Models

Topics over Time (Wang & McCallum 2006) and Dynamic Topic Models (Blei & Lafferty 2006) are more complicated LDA-based models (Blei, Ng, & Jordan 2003) that could also be used to induce topics (events) from blog corpora. These models extend LDA by modeling the time of generation of a document in addition to the contents of the documents which is useful when topics are interpreted as news events since time spans are integral to the notion of an event. However, they treat documents as being generated by mixtures of topics, whereas our model assumes that each document is related to a single topic. We believe that this stronger assumption is appropriate for the short news-driven blog postings that dominate our corpus.

Date	Cluster label
May 4	Obama wins Guam
May 5	IN and NC primaries
May 7	McGovern endorses Obama
May 8	Hillary claims wider support base
May 9	Obama superdelegate lead
May 14	Edwards endorses Obama
May 20	Hillary possible VP pick?
May 21	Kentucky and Oregon primaries results
May 23	Kennedy assassination gaffe
May 30	James Carville backs Obama to win
May 31	Obama resigns from his church

Table 1: Resultant clusters after running SpeCluster Over Time (T=13). The cluster labels above were hand created after inspecting the top 100 documents that were assigned to each cluster.

Results

The results of clustering using the SpeCluster Over Time (SCOT) model are very sensitive to the initialization of the multinomials due to the non-convex function optimized in the EM procedure. This issue is especially evident in the Six Apart blog corpus due to the wide range of topics involved in each blog entry as compared to mainstream news stories. To reduce variance in results caused by random initialization, the experimental results are averaged over 10 runs. Another method adopted to deal with the issue is to initialize the topic distributions with the clusters obtained from clustering the news stories, while performing inference on the SixApart blog corpus. The belief is that the news clusters provide a reasonable starting point. The news and blog corpora are processed independently. Each document (a news article or a blog post) is converted to a term vector. Terms that occur fewer than five times in the corpus are discarded. Table 1 shows hand-created summaries of blog posts in each cluster induced by SCOT (using the news corpus for initialization). The number of clusters is preset to 13 when running the experiments. Results from two clusters were eliminated since they primarily contained documents from non-English blogs. Quantitative evaluation and discussion of these results are provided in a later section.

Finding Timelines with Graph-Clustering Methods

The probabilistic approach based on blog and news text outlined above is one possible approach to timeline construction. In this section we describe methods based on link graph of the blog posts instead of their textual content. One advantage of this family of approaches is that the naturally take advantage of the relational structure of the data (e.g., hyperlinks between postings and news articles).

The link graph-based construction system we will describe is composed of three components. The first component takes the query and a time range from the user as input, interacts with the search engine, and return a ranked list of blog posts relevant to the topic. The second component transforms the blog posts into a graph; the following section describes this in more detail. The third component, given the graph and a link-based clustering algorithm, produces event-clusters with a date and a representative document corresponding to each event.

Transforming Blog Posts to a Graph

Often URL links are found in blog posts; if we see each post as a node and links from one post to another as an edge, we can easily transform a set of blog posts into a graph. However, most blog posts, especially blog posts pertaining to current events, do not contain links to other blog posts; using blog-to-blog links would result in a vary sparse graph.

Instead of linking to other blog posts, blogs often link to news articles found on major news websites such as nytimes.com or washingtonpost.com. Adding each linked news articles to the graph as nodes and the links themselves as edges, we create a denser, mostly bipartite graph on which we can run link-based clustering algorithms.

Time-based Graph Clustering

After the graph is constructed, graph clustering algorithms are used to produce clusters of blog posts and news articles, with each cluster corresponding to an event in the timeline, ideally. This section will describe two graph clustering methods we propose for doing this and will also describe how we use time (publishing date of the blogs) to guide the clustering. Both of these proposed clustering methods are based on random walks on graphs, so we will first briefly describe it below before moving on the to the clustering methods.

Random Walks on Graphs Given a graph $G = (V, E)$, random walk algorithms return as output a ranking vector $\mathbf{r}$ satisfying the following equation:

$$\mathbf{r} = (1 - d)\mathbf{u} + dW\mathbf{r} \quad (2)$$

where W is the weighted transition matrix of graph G where transition from i to j is given by $W_{ij} = 1/degree(i)$. $\mathbf{u}$ is a normalized teleportation vector where $|\mathbf{u}| = |V|$ and $||\mathbf{u}||_1 = 1$. d is a constant damping factor. The ranking vector $\mathbf{r}$ can be solved for by finding the dominant eigenvector of $(1-d)(I - dW)^{-1}\mathbf{u}$ or iteratively substituting $\mathbf{r}^t$ with $\mathbf{r}^{t-1}$ until $\mathbf{r}^t$ converges. Equation 2 can be interpreted as the probability of a random walk on G arriving at node i, with teleportation probability $(1-d)$ at every step to a node with distribution $\mathbf{u}$. For later use we will define the ranking vector $\mathbf{r}$ as a function of G, $\mathbf{u}$, and d: $\mathbf{r} = RandomWalk(G, \mathbf{u}, d)$.

MultiRankWalk Our first proposed clustering method is actually a graph-based semi-supervised learning method. Semi-supervised learning methods are used in classification problems when very few training instances are available. However, we do not have labels training instances and we do not know the number of classes (clusters) that are required for this method. To solve this problem, we use the time information available in blog data (publish date of blog posts) to we create initial *seed clusters* as labeled instances for this algorithm. Details of creating seed clusters can be found after the next section.

After obtaining seed instances, the graph G describes data in a classification learning framework: the nodes are instances and edges represent similarity or relations between the instances. Labeled training instances of each cluster is described by a vector $\mathbf{u}$, the *seed vector*, where each non-zero element corresponds to a seed instance. The random walk describes classification as a process of finding similar instances based on citation or recommendation of the current instance. For each cluster c, at every time step the process may follow a recommendation with probability d or it may decide to start the process again at an instance labeled c with probability $1 - d$. The process is repeated for every cluster and the cluster of an unlabeled instance is decided by which class c's process visited the instance most often. The learning algorithm is formally described in Figure 4.

Given: A graph $G = (V, E)$, corresponding to nodes in G are instances X, composed of unlabeled instances X^U and labeled instances X^L with corresponding labels Y^L, and a damping factor d.

Returns: Labels Y^U for unlabeled nodes X^U.

For each cluster c

1. Set $\mathbf{u}_i \leftarrow 1, \forall Y_i^L = c$
2. Normalize $\mathbf{u}$ such that $||\mathbf{u}||_1 = 1$
3. Set $R_c \leftarrow RandomWalk(G, \mathbf{u}, d)$

For each instance i

- Set $X_i^U \leftarrow argmax_c(R_{ci})$

Figure 4: The MultiRankWalk algorithm.

We will refer to this classification algorithm as **MultiRankWalk**, because it computes **Multi**ple **Rank**ings using random **Walk**s. This algorithm is similar to previously described methods in (Zhou *et al.* 2004; Gyongyi, Garcia-Molina, & Pedersen 2006). Besides producing clusters, this method also ranks every instance within each cluster; this ranking information can be used to produce representative documents or summarization for the event corresponding to the cluster.

K-Walks The proposed K-walks clustering method is very similar to a K-means clustering algorithm but uses random graph walk when calculating distances between nodes. Specifically, for calculating the center of a cluster of nodes it uses PageRank (PR) (Page *et al.* 1998). For calculating the distance between a node and a center it uses personalized PageRank or random walk with restart (RWR) (Haveliwala, Kamvar, & Jeh 2003; Tong, Faloutsos, & Pan 2006).

Input: A weighted transition matrix W, number of clusters k, teleportation probability α, and restart probability β (Here α and β are analogous to the damping effect $1-d$ in Equation 2).
Output: Clusters $C_1, C_2, ...C_k$.

1. Initialize cluster centers $c_1^0, c_2^0, ..., c_k^0$.
2. Set $t = 0$.
3. Obtain walk vectors w_i^t using RW from c_i^t with restart probability β
4. Cluster each point a according to the walk vectors, where $a \in C_*^{t+1}$ if $* = argmax_i w_i^t(a)$.
5. Obtain new centers c_i^{t+1} using RW with teleportation probability α using the subgraph formed by nodes in C_i.
6. If not converged, go to Step 3 and set $t = t+1$; otherwise, stop.

Figure 5: The K-walks algorithm.

Unlike MultiRankWalk, K-walks is an unsupervised clustering algorithm and can readily be used on the graph. However, like original K-means algorithm, there are two issues: first, we need the number of clusters k, and second, poor initial cluster centers can result in poor clustering. So instead of an arbitrary k and randomly

choosing initial centers, we use the time information available in blog data to help us choose k and the initial cluster centers, the details of which is described in the next section.

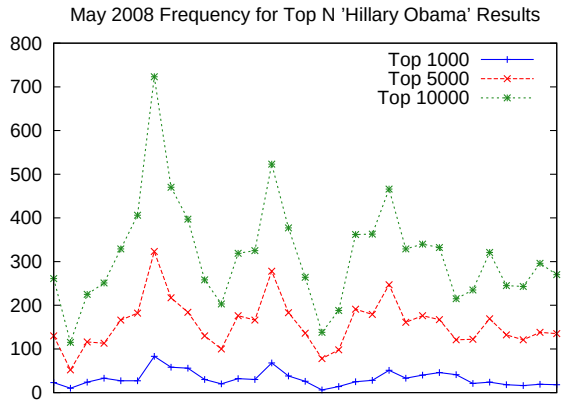

Figure 6: A frequency chart of top N posts in May 2008 returned by the search query "Hillary Obama." The x-axis represents the days of the month and the y-axis represents the number of posts. A point indicate the number of posts posted on a particular day.

Seeding and Guiding Clustering Using Time To create initial seed clusters for specifying the number of cluster and guiding the clustering algorithms, we make two simplifying assumptions: 1) the blog posts published around the same time are more likely to be about the same event, and 2) only one major event happens at one discrete time unit in the timeline. With these two assumptions in mind, we take the blog posts returned by the search query and plot the number of posts published against a discrete time unit, which, in this work, we decide to be a 24-hour period—a day. From this plot we can then define *peaks*: a *peak* occurs when the number of posts published on a time unit is greater or equal to the number of posts published in the previous time unit and the following time unit. Using this definition, we set the number of clusters to be the number of peaks and the seed instances of a cluster to be all posts published within the time unit of the corresponding peak. Figure 6 shows a frequency chart of top posts in May 2008 returned by the search query "Hillary Obama"; the *peaks* seen in the chart corresponds to major events during in the 2008 U.S. Democratic presidential primary.

Results

In this section we take a look at the result of the K-walks graph clustering method. For this specific corpus and topic, the result of MultiRankWalk is almost the same as that of K-walks, so for sake of space it is not shown here. The top 1000 blog posts returned by the search engine with the query "Hillary Obama" are used to construct the graph. After filtering out pointer posts (posts with more than five links and no content), duplicate posts, and posts that are not co-linked to another post, the remaining posts and articles linked by the posts are transformed in to a graph of roughly 300 nodes. The seeding method described above is used to determine the initial seeds and the number of clusters, and we use the conventional restart and teleportation factor $\alpha = \beta = 0.15$ for random walk parameters. The resulting blog post clusters are examined by human and hand-assigned a label or labels as to which event(s) each cluster contains, shown in Table 2. For automatically generated view of the clusters, please see the appendix.

Date	Cluster label
May 4	Obama wins Guam, "obliterate Iran" remark
May 8	Hillary claims wider support base
May 9	Obama superdelegate lead
May 14	Edwards endorses Obama
May 21	Hillary's soaring debt, Hillary possible VP
May 24	Kennedy assassination gaffe

Table 2: Resultant clusters after running K-walks. The cluster labels above were hand created after inspecting the top 10 blog posts that were assigned to each cluster.

The result reveals that some clusters contain at least two events that happened around the same time. For example, the May 4th cluster contains the events "Obama wins Guam" and "'Obliterate Iran' remarks." The May 21st cluster also contains two events. This shows that link structures in the top documents definitely contain event clusters, but determining the number of cluster by using post frequency analysis may be too coarse-grained to differentiate events that happen around the same time. Quantitative evaluation and discussion of these results follow in the next section.

Quantitative Evaluation

Hand-produced timelines

To evaluate these results, three of the authors hand-produced timelines for Democratic primary subcorpus that indicated the most important events of May 2008. The timelines were fairly minimal, consisting of a description and a timespan for each event. In addition, events were linked by an *inclusion* relationship, as described below. The timelines were produced independently (i.e., without consultation between annotators), and the annotators were encouraged to use their background knowledge of the domain, as well as examination of the corpus, in preparing the timeline.

We expected that disagreements would arise from several different sources. Most obviously, there are many ways to describe the same event: e.g., "John Edwards announces endorsement of Barack Obama" versus "Edwards backs Obama." Another type of possible disagreement concerns which events are "most important": e.g., some annotators might consider the Guam Democratic caucus important, while others might not. Yet another type of disagreement concerns the *granularity* of events: e.g., one annotator might produce a single event "Obama does better than expected in Indiana and NC primaries" while another might produce two distinct events, "Obama wins North Carolina" and "Clinton wins Indiana'.

Date	A1	A2	A3	Event
May 3	X	X		Obama wins Guam
May 4		X	X	"Obliterate Iran" remarks
May 6-7	X	X		IN and NC primaries
May 9-10	X	X		Obama superdelegate lead
May 13	X	X	X	WV primary
May 14-15	X	X	X	Edwards endorses Obama
May 19-20	X	X		OR, KY primary
	X		X	Obama pledged delegate lead
May 23	X	X	X	Kennedy assassination gaffe

Table 3: Events from hand-produced timelines that were selected as important by two or more human annotators

	A1	A2	A3	Avg	2+
A1		0.84	0.56	0.70	
A2	0.84		0.52	0.68	
A3	0.56	0.52		0.54	
k-walks	0.44	0.61	0.48	0.51	0.41
SpeCluster					
−init,−prior	0.58	0.55	0.40	0.51	0.71
+init,−prior	0.54	0.48	0.34	0.46	0.68
−init,+prior	0.74	0.84	0.54	0.74	0.66
+init,+prior	0.76	0.80	0.57	0.71	0.61

Table 4: Pairwise agreement between annotators and algorithms. The 2+ column shows the agreement between the method output the *consensus events* shown in Table 3, and +/-init indicates whether or not the news data was used to initialize the SpeCluster method, and +/-prior indicates the presence or the absence of the inverse gamma prior. In the case of -init, the results are averaged over 10 runs with random initialization

Finally, annotators might disagree on the very definition of an "event." In many cases, clusters of text are related to events that inarguably take place in the "real world" (e.g., primary elections); however, it is also possible to have clusters of blog postings that are initiated by postings from influential bloggers, pundits, or political figures. Below, we will call these *discourse events*. One problem is that for our sample task, it is unclear how to clearly separate discourse and non-discourse events definitionally, since there is no underlying clear separation between the world of discourse and the world of politics.

Of course, all of these sorts of inter-annotator disagreement may also arise in comparing human-provided annotations with computer-generated annotations. In order to control for, and potentially measure, the contribution of these various sources of disagreement, we extended the annotation task in two ways. First, annotators were asked to mark events as "discourse events" when they felt this was appropriate. Second, annotators were encourages to record events at various levels of granularity, and to indicate when a more abstract event included one or more more concrete events: hence the human timelines were actually event trees, rather than linear sequences of events.

The hand-produced timelines had between 16 and 21 events, with a fairly large amount of variation between annotators: only nine events were selected by more than one annotator, and only 3-4 were selected by all three. Table 3 summarizes the overlap between the annotators.

Tree Edit-Distance Evaluation

To more quantitatively measure agreement, we adopted a variation of an approach widely used in computational biology: in particular we wrote code to align two event trees s and t by finding the minimal sequence of "edits" that will transform s into t. We ignore the text descriptions for events, and only attempt to align the times. We assume that s and t are both sequences of event trees, which are of depth at most two (i.e., are primitive events, or abstract events with primitive events as children), and allow two three operations: deletion of an event tree, modification of the time of an event tree, or replacement of a depth-two tree with its children. Replacement has cost zero, and deletion has cost 1 for top-level events and primitive events, cost 0.1 for second-level events, and deletion cost is decreased by an additional factor of 0.5 for discourse events. Modifying a time (either a start-time or an end-time) has cost $0.01 \cdot 10^k$ for a k-day change–i.e., the cost is only 0.1 for a one-day change, but 1 for a two-day change, and 10 for three-day change.[2] Finally, to account for the effect of varying length, this edit distance cost is normalized by the cost of deleting every event in both s and t, and subtracting the result from 1.0. This yields an agreement measure between 0.0 and 1.0 (where 1.0 indicates a perfect alignment, and 0.0 indicates that no event from s can be usefully aligned with any event in t).[3]

Inter-annotator agreement with this measure averages 0.64 for the three pairs of annotators, with agreement values ranging between 0.84 and 0.52.

For the k-walks timeline, agreement to the human annotators averages 0.51, with a minimal values of 0.44 and a maximum value of 0.61.

The results of the initial probabilistic clustering were harder to interpret, because they did not form a tree—instead the clustering algorithm produced two clusters (e.g., "primary results for any state") that were highly coherent topically, but not temporally compact. If these clusters are manually deleted (so that the result can be automatically tree-aligned with human-produced timelines) the average agreement is 0.46, with a minimum of 0.34 and a maximum of 0.54.

Alternatively, the algorithm can be modified to discourage production of such clusters, by imposing a prior on the variance of times for each cluster. If this is done, the average agreement improves to 0.71, with a minimum value of 0.57 and a maximum value of 0.80—a result slightly *better* than the average intra-annotator agreement. Surprisingly, we notice that initializing the EM procedure with the results of clusters obtained from the news corpora yields worse results than initializing randomly in all cases but one.

These results are summarized in Table 4, which also shows the agreement between the automatic methods and the consensus event sequence of Figure 3.

Discussion

In general, the automatic methods were able to provide coherent clusters and their consensus agreement with human annotators are close to inter-annotator agreement and in some cases event rivals inter-annotator agreement.

SpecCluster Over Time Compared to K-walks, SCOT provides higher agreement with human annotators, and in some cases SCOT's agreement with human annotators rivals even agreement between human annotators. One aspect of the experiment that contributed to the superior performance of SCOT is the hand picking of number of clusters and the elimination of non-English clusters in SCOT (whereas in K-walks the number of clusters were estimated automatically). Another unexpected result from SCOT is the comparison between +init and -init: the initialization of the EM procedure with results of the news clustering were poorer than random initialization. One possible explanation for this could be the non representative nature of the news corpus used (a post-experiment

[2]This schedule was based on the observation that annotators rarely disagree by more than one day.

[3]One disadvantage of this approach is that, since the description of events is ignored in computing edit distance, two distinct events that happen to occur at the same time may be incorrectly aligned. We will ignore this issue in the discussions below; however, manual inspection of aligned events suggest this sort of mismatch is rare for event trees with high agreement, and more common for event trees with low agreement, suggesting that the effect is unlikely to change the relative ordering of agreement between two techniques.

examination of the news clusters shows that the corpus is not as primaries-focused as expected). Note also that results of -init runs have high variance and the results in Table 4 are averaged over 10 runs.

K-walks A couple of factors are likely to have contributed to the lower performance of K-walks. First, the sparse linkage information found in this dataset (about 150 linked blog posts nodes out of a corpora consisting of 1000 documents), which resulted in a rather sparse graph that included many small disconnect components that are bad for graph-clustering. Second, K-walks produced far fewer event-clusters than human annotators; this is a shortcoming of the post-frequency analysis, which determines the number of clusters, rather than a problem inherent to the graph-clustering methods. One interesting observation is that several German documents were clustered correctly, demonstrating one likely advantage of link analysis methods, that they are language-independent. Note that even though only 150 out of 1000 blog posts retrieved by query "Hillary Obama" were clustered using this method, it is not necessary less useful in constructing a timeline and present exemplar documents for each event.

Conclusions

In this paper this we addressed the task of producing explicit representations of "sagas". To do this we considered two alternative unsupervised methods - one based on probabilistic language models, and one based on network analysis. We evaluated these both qualitatively (in Tables 1 and 2) and quantitatively, by measuring agreement with human-provided annotations.

Quantitatively, both techniques are broadly comparable to human-produced annotations; however, this is true partly because human agreement is relatively low for this task. This suggests further study of "timeline completion" tasks, in which more information is provided by the user; such semi-automatic approaches may produce timelines that better agree with the goals of a particular user. Another possible avenue for future research is creating a timeline using multiple types of social media (e.g., weblogs and netnews); intuitively events should be reflected redundantly in both media types.

References

Blei, D. M., and Lafferty, J. D. 2006. Dynamic topic models. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, 113–120. New York, NY, USA: ACM.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research* 3:993–1022.

Fraley, C., and Raftery, A. E. 2007. Bayesian regularization for normal mixture estimation and model-based clustering. *J. Classif.* 24(2):155–181.

Gyongyi, Z.; Garcia-Molina, H.; and Pedersen, J. 2006. Web content categorization using link information. Technical report, Stanford University.

Haveliwala, T.; Kamvar, S.; and Jeh, G. 2003. An analytical comparison of approaches to personalizing pagerank. Technical report, Stanford University.

Huang, Y. 2006. Text clustering with extended user feedback. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 413–420. SIGIR.

Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1998. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project.

Tong, H.; Faloutsos, C.; and Pan, J.-Y. 2006. Fast randomwalk with restart and its applications. In *Proceedings of the 2006 IEEE International Conference on Data Mining (ICDM 2006)*.

Wang, X., and McCallum, A. 2006. Topics over time: a non-markov continuous-time model of topical trends. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 424–433. New York, NY, USA: ACM.

Yang, Y.; Pierce, T.; and Carbonell, J. 1998. A study of 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*, 28–36. New York, NY, USA: ACM.

Zhou, D.; Bousquet, O.; Lal, T. N.; Weston, J.; and Scholkopf, B. 2004. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*.

Appendix: Automatically Generated Event Descriptions

Table 5 shows the most probable words of the multinomials belonging to some key topics induced by the SpeCluster Over Time model. Table 6 shows the top terms in clusters using K-walks according the term's TF-IDF score.

Topic	Most probable terms
General	obama, hillary, clinton, he, has, barack, about, mccain, said, john
1	guam, obama, democracy, closing, debate, fantastic, gov, ron, paul, affairs
2	obama, clinton, has, superdelegates, hillary, democratic, up, delegates, stae
3	americans, white, clinton, hillary, working, support, weakening, black, usa, tax
4	obama, super, wright, his, rep, dnc, public, ayers, claim, support
5	obama, hillary, she, kennedy, june, assasination, california, campaign, my, husband

Table 5: Top terms from multinomials induced in the SpeCluster Over Time model

Date	Top Post Title Terms
May 4	guam, lead, option, tutorial, 1, nuclear, x, hillary, obama, s, gaat, afp, heat
May 8	wtf, hillary, clinton, stupid, sinks, andrews, plays, people, didja, backs
May 11	victory, obama, literally, americans, bus, remark, moves, who, kumar, virginia
May 14	edwards, sds, vote, switches, leads, michelle, abc, drama, endorsement, mom
May 21	search, million, told, 31, wanted, strike, debt, v, percent, begins, way, 50, vp
May 24	assassination, apologizes, comment, f, robert, remarks, kennedy, out, hillary

Table 6: A clustering result using K-walks on the top 1000 results returned by the search query "Hillary Obama." The top terms are calculated using TF-IDF term weighting.