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

News interfaces are largely driven by recent information, even if many events are better interpreted in context of previous events. To address this problem, we consider the task of constructing an explicit representation of a "saga"—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 hand-produced timelines using a tree edit distance measure. Preliminary results using these techniques on a weblog corpus and a supplementary news corpus are presented, showing both promise and challenges.

Introduction: Why Timelines Are Useful

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, and many events are better interpreted in context of previous related events.

The hypothesis behind our work is that it is useful to construct representations of such "sagas"—i.e., long-running sequences of related events. From an application perspective, we are interested in providing tools giving a reader the complete narrative context of a given event. From a sociological perspective, we are interested in finding out how events are perceived, reported and synthesized in a number of media types including news and weblogs.

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. The edges in an event graph encode binary relations between events. These relations could be simple temporal relationships but could also capture causality or other deeper relationships. 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. In this work we will mine timelines from weblogs. One potential advantage of 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.

Challenges in Constructing Timelines

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. We will begin by proposing a precise notion of a *timeline*.

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.

Given a particular corpus C of documents, 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.

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: 1) a short textual description q_i; 2) an associated real-world event e_i; 3) a *time span* t_i indicating the duration of e_i, where t_i is further defined by a start and end time; 4) 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; and 5) a sample S_i^C of highly-relevant documents from C.

We use the term *timeline completion* for the task of 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. If only C is given, then finding r_i^C is an unsupervised clustering problem, which we will focus on in the remainder of this paper.

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. For the experiments described below, we focus on a subset of the corpus related to the US Presidential Democratic primaries. The top 1000 or 2000 documents returned by issuing the query "hillary obama" were used. The news corpus comprising of the news stories linked to from the blogs is also used in generative model 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. 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.

A maximum likelihood estimate of the parameters that maximizes the likelihood of observing the corpus can be obtained by running an EM procedure.

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 Using Graph Clustering

In this section we describe methods based on link graph of the blog posts instead of their textual content. 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 and returns a ranked list of blog posts relevant to the topic. The second component transforms the blog posts into a graph. 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.

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 transform a set of posts into a graph. However, many blog posts do not contain links to other blog posts, resulting in a sparse graph. We find that instead blogs often link to news articles, and if two or more posts link to the same arti-

Input: Transition matrix W, number of clusters k, teleportation and restart probability α, β.
Output: Clusters $C_1, C_2, ...C_k$.

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

Figure 1: The K-walks algorithm.

cle, we add these links and articles as edges and nodes in the graph to create a denser, mostly bipartite graph on which we can run link-based clustering algorithms.

Time-based Graph Clustering

The proposed graph clustering method is based on random walks on graphs, so we will first briefly describe it below before moving on the to the clustering methods. Given a graph $G = (V, E)$, random walk algorithms return as output a ranking vector $\mathbf{r}$ satisfying the following equation: $\mathbf{r} = (1 - d)\mathbf{u} + dW\mathbf{r}$ 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.

K-Walks The K-walks clustering method we propose here is very similar to the K-means clustering algorithm but uses random graph walk probabilities as 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 random walk with restart (RWR) (Haveliwala, Kamvar, & Jeh 2003; Tong, Faloutsos, & Pan 2006).

Seeding and Guiding Clustering Using Time K-walks is an unsupervised clustering algorithm and can readily be used on any 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 are described in the next section.

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—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.

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.

Results

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 colinked to another post, the remaining posts and articles linked by the posts are transformed in to a graph of roughly 300 nodes. 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 corresponds to, shown in Table 2.

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." Yet another type of disagreement concerns the *granularity* of events. 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.

	A1	A2	A3	Avg
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
SpeCluster				
$-$init,$-$prior	0.58	0.55	0.40	0.51
$+$init,$-$prior	0.54	0.48	0.34	0.46
$-$init,$+$prior	0.74	0.84	0.54	0.74
$+$init,$+$prior	0.76	0.80	0.57	0.71

Table 3: Pairwise agreement between annotators and algorithms. +/-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

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, annotators were encourages to record events at various levels of granularity, and to indicate when a more abstract event included one or 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.

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; the details of the algorithm are found in the appendix.

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 algorithm produced two clusters that were highly coherent topically, but not temporally compact. If these clusters are manually deleted the average agreement is 0.46, with a minimum of 0.34 and a maximum of 0.54. Alternatively, the algorithm can be modified by imposing a prior on the variance of times for each cluster. Doing so improves the average agreement 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 3.

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.

References

Blei, D. M., and Lafferty, J. D. 2006. Dynamic topic models. In *ICML 2006*.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *JMLR* 3:993–1022.

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 *SIGIR 2006*.

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 *ICDM 2006*.

Wang, X., and McCallum, A. 2006. Topics over time: a non-markov continuous-time model of topical trends. In *KDD 2006*.

Appendix: Tree Edit-Distance

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