TimeMachine: Timeline Generation for Knowledge-Base Entities

Tim Althoff*, Xin Luna Dong†, Kevin Murphy†, Safa Alai†, Van Dang†, Wei Zhang†

*Computer Science Department, Stanford University, Stanford, CA 94305
†Google, 1600 Amphitheatre Parkway, Mountain View, CA 94043

*althoff@cs.stanford.edu †{lunadong, kpmurphy, safa, vandang, weizh}@google.com

ABSTRACT

We present a method called TIMEMACHINE to generate a timeline of events and relations for entities in a knowledge base. For example for an actor, such a timeline should show the most important professional and personal milestones and relationships such as works, awards, collaborations, and family relationships. We develop three orthogonal timeline quality criteria that an ideal timeline should satisfy: (1) it shows events that are *relevant* to the entity; (2) it shows events that are *temporally diverse*, so they distribute along the time axis, avoiding visual crowding and allowing for easy user interaction, such as zooming in and out; and (3) it shows events that are *content diverse*, so they contain many different types of events (*e.g.*, for an actor, it should show movies and marriages and awards, not just movies). We present an algorithm to generate such timelines for a given time period and screen size, based on submodular optimization and web-co-occurrence statistics with provable performance guarantees. A series of user studies using Mechanical Turk shows that all three quality criteria are crucial to produce quality timelines and that our algorithm significantly outperforms various baseline and state-of-the-art methods.

Categories and Subject Descriptors: H.2.8 **[Database Management]**: Database applications—*Data mining*

General Terms: Algorithms, Experimentation.

Keywords: Summarization, Timeline, Knowledge Base, Submodular Optimization.

1. INTRODUCTION

As the web and other technological advancements continue to bring down barriers for creation and distribution of information, relevant information is often buried in an avalanche of data, and locating it has become increasingly difficult [30]. Search engines have attempted to address this challenge [4], but the volume and diversity of results can still be overwhelming, even for simple entity queries [31]. In many such cases, for instance when searching for a person or organization, an overview of the most important events in an organized and readable format would serve users better, ideally with interactive features to enable further exploration. A *timeline* with clickable key events arranged along a horizontal time axis would serve this need [39].

KDD'15, August 10-13, 2015, Sydney, NSW, Australia.
ACM 978-1-4503-3664-2/15/08.
DOI: http://dx.doi.org/10.1145/2783258.2783325 .

Automatically generating timelines is very challenging. To be specific, consider creating a timeline for the American actor Robert Downey Jr. There are hundreds of possible candidate events and it is infeasible to display all of them. Robert Downey Jr. is best known for his starring roles in the movies *Iron Man* and *Avengers*, but even for a single movie there are dozens of related events to display (production, release dates, opening, and award ceremonies). In fact, one should not only focus on movies but provide a more holistic overview of his life and career. This could include showing various family relationships (*e.g.*, father Robert Downey Sr., ex-wife Deborah Falconer, or wife Susan Downey), important acting roles for his career (the movie *Chaplin* and TV show *Ally McBeal*), and other notable works and professional relationships. However, note that events might be related as well — if one includes a movie award one might not want to display its release date separately but rather show a more diverse event instead. Lastly, the timeline should be interactive to enable further exploration.

Knowledge bases (KB) of timestamped facts such as Freebase [6] or YAGO [35] have been used as the source of event information (in this paper we use Freebase). Previous work has introduced timeline generation from KBs through visualizing entity-level co-occurrence in news corpora [26], displaying events associated with an entity in YAGO [40], and generating context-aware timelines from Wikipedia [39]. However, these works did not address the problem of selecting a subset of events but instead displayed all events [26, 40], or have used a static global ranking that does not capture dependencies between events and is therefore unable to encourage diversity [39]. Furthermore, this existing work has not considered challenges raised by enabling user interaction nor provided an empirical evaluation of the quality of the generated timelines.

Present work. In this paper, we develop an approach called TIMEMACHINE to generate a timeline for a given entity of interest. We develop three orthogonal timeline quality criteria:

1. Relevance: Display only the most "interesting" or "relevant" events in an entity's history.
2. Temporal Diversity: Distribute events evenly along the temporal axis, to avoid visual crowding, and to allow easy interaction with the depicted events.
3. Content Diversity: Display a diverse set of event types (*e.g.*, for an actor, do not only list the movies they have been in).

Consequently, we propose a principled solution to timeline generation according to these criteria based on submodular optimization, for which we both provide theoretical performance guarantees and show empirical evidence of significant improvement over baseline and state-of-the-art methods.

In Figure 1, we show that our approach successfully generates a timeline of relevant events that is diverse both in terms of content and time. This timeline is also interactive in three ways. First,

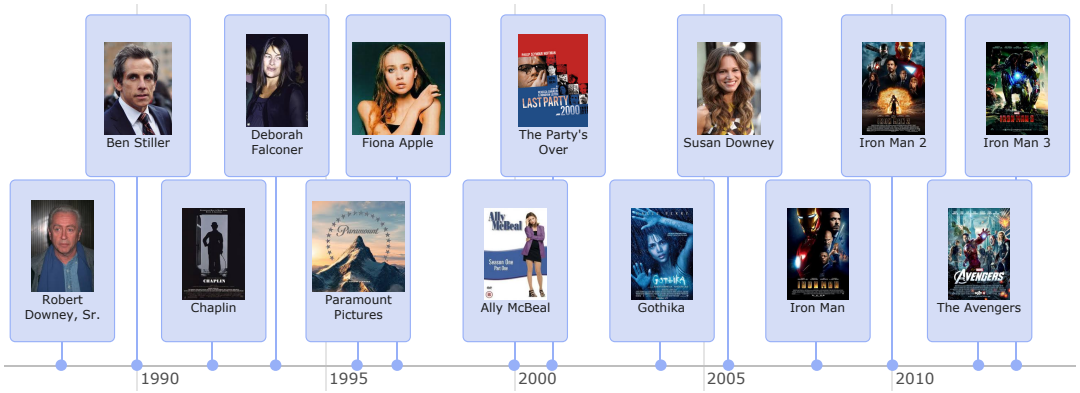

Figure 1: An example timeline for Robert Downey Jr. (American actor) generated by our proposed approach. Note that the timeline is interactive and displays explanations for each event on hover (see Figure 4). Furthermore, it can be dynamically zoomed.

when the user hovers over an image, we show various details, such as "Robert Downey Jr. starred in *The Avengers*, released on April 11, 2012". Second, the user can specify a particular time period, and a new timeline for that period will replace the current one. For example, Figure 4 shows the timeline for Robert Downey Jr. from 2007 until 2014 and gives an example event description. The zoomed-in version focuses attention on more recent events, such as his award for *Tropic Thunder* and his role in the *Sherlock Holmes* movies. Finally, the user can click on an entity icon, such as Susan Downey, and a timeline for this entity will be displayed.

Our approach involves the following two main steps, which are sketched in Figure 2. First, given a subject entity of interest, we generate as many candidate events as possible by searching for neighboring entities with timestamps in the given knowledge base (Section 2). We generate candidate events for all possibly interesting subjects offline. Second, given a set of candidates and a time period of interest, we select (online) a diverse subset of the most relevant events subject to temporal diversity or layout constraints (Section 3). To do this, we maximize a submodular objective using various relevance signals based on web co-occurrence, subject to these layout constraints. We prove that our greedy algorithm for optimization yields close-to-optimal solutions. In addition, our algorithm allows for fast dynamic updates of the timeline based on user interaction (zooming in or out).

We evaluate our proposed algorithm through a series of user studies with 1154 raters and compare it to various simpler baselines and state-of-the-art approaches (Section 4). Our experiments show that users always significantly prefer our proposed method (60-91% of timeline comparisons). Further, we demonstrate that enforcing temporal diversity and content diversity significantly improves the results.

In summary, our main contributions are as follows:

1. A design of a timeline search engine that efficiently supports various types of user interaction.
2. An algorithm for generating entity timelines based on submodular optimization and web-co-occurrence scores.
3. An extensive user study of the relative importance of different signals for determining entity relevance and different notions of diversity.

2. EVENT CANDIDATE GENERATION

Recall from Figure 2 that there are two main steps: candidate event generation and event selection. In this section, we describe how we generate candidate events given a subject of interest. Our approach is to generate a large set of events, and then to filter out "irrelevant" ones. We give an evaluation of this filtering step. This

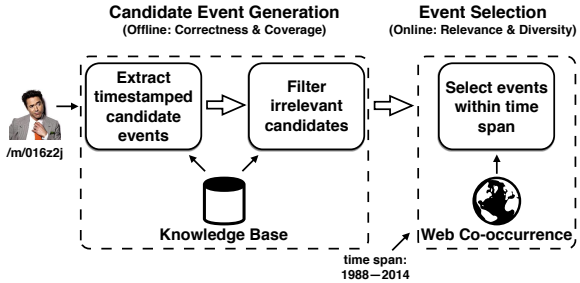

Figure 2: System architecture. TIMEMACHINE **traverses the KB offline to generate candidate events for a subject of interest (*e.g.*, Robert Downey Jr.). At run time, the user specifies a time period of interest and** TIMEMACHINE **selects a subset of events from the candidates to generate the timeline.**

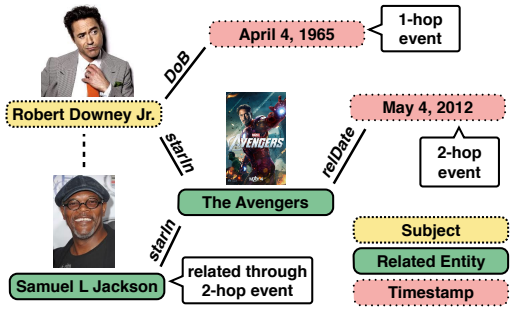

Figure 3: An illustration of the event candidate generation step. Events are short paths that are associated with a timestamp.

is a necessary preparation step for our key contribution in this paper (described in the following section): dynamically selecting a subset of the remaining events at runtime, depending on the time span of interest and the available screen real estate.

2.1 Event Generation

We can consider the KB as a graph with nodes representing subjects and objects, and edges representing the relationship (predicate) between the nodes. Given a particular subject represented by a node N_s in the KB, we are interested in nodes that are connected to N_s through some paths and are associated with a timestamp; we call such paths "events". As we discuss below, we consider two kinds of events: simple and compound. Figure 3 illustrates the overall process for Robert Downey Jr.

Simple events. Simple events are nodes with timestamps that can be reached by paths of length one or two starting at N_s. In the

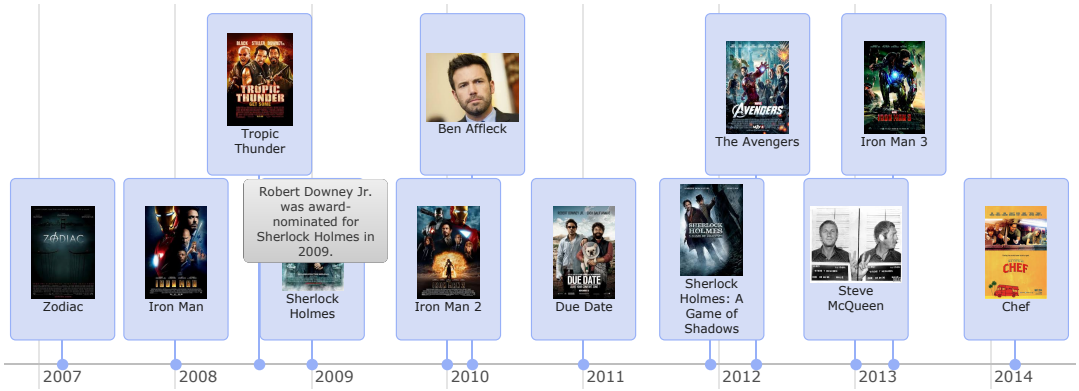

Figure 4: An example timeline for Robert Downey Jr., where we have zoomed in on the most recent years of his life. Note the description for the Sherlock Holmes event (award nomination) that gets displayed on hover.

example in Figure 3, we can traverse along an edge of type *date-of-birth*, and reach a node representing the corresponding date; this is a path of length one (1-hop event). We can also traverse along an edge of type *starred-in*, and reach the node representing the movie *The Avengers*. To find the corresponding date, we traverse along a second edge of type *release-date* and reach a node with the release date of the movie. This is a path of length two.

We can formally represent a simple event (derived from a path of length two) as follows: $s \xrightarrow{p_1} re_1 \xrightarrow{p_2} t$, where s is the subject, re_1 is a related entity (such as *The Avengers*), t is the timestamp, and p_1 and p_2 are the predicates along the path. For simplicity, we represent 1-hop events in a similar way, by introducing a self-loop through $p_1 = self$ and $re_1 = s$. For each event e, we define the *subject* of the event as $\text{SUB}(e) = s$, the *related entity* as $\text{RE}(e) = re_1$, the *timestamp* as $\tau(e) = t$, the *entity path* as $\pi_{re}(e) = p_1$, and the *time path* as $\pi_t(e) = p_1.p_2$.

Compound events. Extracting only simple events will miss out on some implicit connections to other related entities. For example, consider the collaboration between Robert Downey Jr. and Samuel L Jackson in the movie *The Avengers* shown in Figure 3. Even though there is no direct edge between Robert Downey Jr. and Samuel L Jackson, they are connected through a path as they starred in the same movie.

We can discover such connections as follows. Suppose we have simple events $s_1 \xrightarrow{p_1} re_1 \xrightarrow{p_2} t$ and $s_2 \xrightarrow{p_3} re_1 \xrightarrow{p_2} t$, which share the same related entity and timestamp, but differ in their first hops. We join these events to generate a new compound event $e = \begin{matrix} s_1 \xrightarrow{p_1} \\ s_2 \xrightarrow[p_3]{} \end{matrix} re_1 \xrightarrow{p_2} t$. We treat s_2 as another related entity re_2 for s_1, and vice versa; in other words, $\text{RE}(e) = re_2 = s_2$, and $\pi_{re}(e) = p_1.p_3$. For a discussion of implementation details please refer to the appendix of the full version of this paper [3].

Event descriptions. To ensure an event can be understood by an end-user when they hover over the corresponding box on the timeline, we have to convert these paths into natural language form. We do this by manually defining some templates for the 100 most frequently occurring paths (see Figure 4 for an example). For the remaining paths, we simply concatenate the English names of the corresponding predicates and entities.

2.2 Event Filtering

The event generation steps we have just described may generate some irrelevant events. For example, it can discover a path *"nationality → date founded"*, so everyone with nationality *USA* has a candidate event with timestamp *July 4, 1776*, the date on which the USA was founded. However, arguably this event is irrelevant to most people, since it is not specific to them, and it occurred well before many of them were born.[1] We propose two simple heuristics that capture these intuitions and filter out many irrelevant events.

The *Frequency Filter* uses the concept of *inverse document frequency* [4] from the IR community. The idea is that an event that is commonly associated with a large number of subjects is unlikely to be particularly interesting. To formalize this, consider the set of all events (s, re, π_t, t). Let $N(\pi_t, re, t)$ be the number of subjects that are connected to re and t through path π_t, and let $N(\pi_t)$ be the number of distinct (re, t) pairs that are connected to any subject via π_t. Furthermore, let $C(\pi_t) = |\{(re, t) : N(\pi_t, re, t) > \theta_1\}|$ be the number of (re, t) pairs for which there are more than θ_1 subjects connected through path π_t. Then for any given path π_t, if $C(\pi_t)/N(\pi_t) > \theta_2$, where θ_2 is some threshold, we drop that path for all subjects. Note that this will generalize across entities. For example, discovering that *"nationality → date founded"* is a irrelevant path based on people born in some countries allows us to drop instances of this path also for people born in all other countries.

Further, entities in a KB are naturally associated with a period of *existence*: individuals are born and pass away, companies get founded and go out of business, and musical groups get formed and split up. The second filter, *Existence Filter*, filters out events that occurred before an entity began to exist. If we find that a particular kind of path is filtered out for a large fraction (say more than θ_3) of entities, we filter the path out for all entities. A canonical example is *"parent → date of birth"* which obviously occurs before the subject entity is born (*i.e.*, $\theta_3 = 100\%$). Based on our experiments (discussed next), we chose $\theta_1 = 50$ and $\theta_2 = \theta_3 = 0.5$, and we observed that slightly varying the parameters had very little impact on the results.

2.3 Evaluation of Event Filtering

We used Freebase [6] to generate candidate events for four types of entities: music artists, actors, politicians, and athletes. We generated candidate events for all entities of these types in Freebase and evaluated the quality of the results.

We evaluated the quality of our filtering using *true positive / negative rate metrics* as follows. First, for each filtering heuristic, we estimate the fraction of filtered paths that were correctly filtered (*i.e.*, judged irrelevant by a human) or the true negative rate. Second, we estimate the fraction of non-filtered paths that are correctly not filtered (*i.e.*, judged relevant by a human) or the true positive rate. For each metric, we evaluated the top 100 most frequent path types covering over 90% of all generated event instances (out of

[1]Even for George Washington, a founding father of the USA, it is safe to eliminate the *"nationality → date founded"* path, as there are other paths connecting him to the USA and its foundation date.

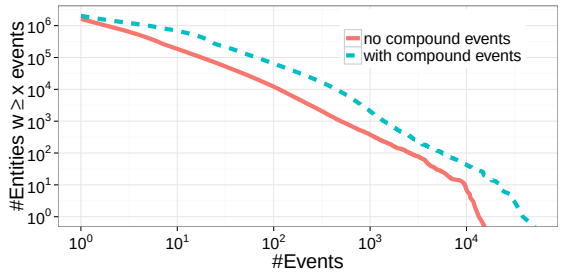

Figure 5: Log-log coverage plot showing the number of entities with X or more candidate events and illustrating the impact of adding compound events (dashed line).

5269 different path types generated in total). Two domain experts manually judged each path as relevant or irrelevant.

We observe that the Frequency Filter is 84% correct (*i.e.*, it accidentally filters out only 16% of the relevant paths), and the Existence Filter is 100% correct (*i.e.*, everything it filters out is irrelevant). The main failure case for the Frequency Filter consisted of relevant events involving many entities, such as large award ceremonies or military conflicts. We also observe that among the paths that pass both filters, 87% are correct. The main failure case are birthdates of related people (*e.g.*, members of the same band), which are arguably irrelevant to the subject.

In addition to high correctness, we need the event generation phase to have high *coverage*. Figure 5 plots the number of events on the X-axis versus the number of entities for which we extracted at least this many events on the Y-axis (after filtering). Suppose we require at least 100 candidate events for an entity before we consider it to be "history rich" enough for us to generate its timeline. The figure shows that we can generate timelines for 12k entities if we use simple events, and for 64k entities if we use compound events (see Section 2.1). This shows that Freebase has a sufficiently rich set of events to make our approach possible, even though it is incomplete in many other ways [12]. With the advent of systems for automated knowledge-base population such as Knowledge Vault [12], we can expect the coverage to improve further in the future.

3. EVENT SELECTION

We showed in the previous section that a given entity may have hundreds of candidate events associated with it. In this section, we discuss our main contribution, a way of selecting a small subset of events to be shown on the timeline, given the time span of interest and a specified amount of space on the screen.

Our approach will be based on submodular optimization, which we explain in general terms in Section 3.1 (following [7, 20]). We define our specific optimization problem in Section 3.2, and give details in Section 3.3 and Section 3.4. In Section 3.5, we describe our efficient approximation algorithm, which we prove in Section 3.6 to yield close-to-optimal solutions. Finally in Section 3.7, we discuss how our algorithm enables user interactions, such as zooming in (or out) on the timeline.

3.1 Submodular Function Maximization

Suppose we have a set (*e.g.*, events) denoted by X, and an evaluation function for sets $f : 2^X \to \mathbb{R}$. Let $f_S(e) = f(S \cup \{e\}) - f(S)$ be the *marginal gain* of adding element $e \in X$ to set S.

A function $f : 2^X \to \mathbb{R}$ is *submodular* if for every pair of subsets $A \subseteq B \subseteq X$ and element $e \in X \setminus B$ we have $f_A(e) \geq f_B(e)$. Intuitively this means that the benefit of adding element e to the smaller set A is bigger than adding it to the bigger set B, so f exhibits the property of *diminishing returns*. We restrict our attention to *monotone* functions; that is, $f(A) \leq f(B)$ for all $A \subseteq B$. We also assume $f(\emptyset) = 0$; that is, f is non-negative.

Constraints. We want to compute $\max_{S \subseteq X} f(S)$ subject to some constraints on S. A common constraint is on the size or cardinality of S [20]. However, in our case, we have more complex constraints, related to temporal diversity and overlap. To formalize these constraints, we need the notion of an independence family, defined as follows. An *independence family* $\mathcal{I} \subseteq 2^X$ is a family of subsets that is downward closed; that is, $A \in \mathcal{I}$ and $B \subseteq A$ implies that $B \in \mathcal{I}$. A set A is called *independent* if $A \in \mathcal{I}$. Popular independence families include *matroids* and intersection of matroids [7]. As an example, given $X = \{a, b, c\}$, an independence family is as follows: $\mathcal{I} = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, c\}\}$.

p-system. For a set $Y \subseteq X$, a set J is called a *base* of Y if J is a maximal independent subset of Y; in other words $J \in \mathcal{I}$ and for each $e \in Y \setminus J$, we have $J \cup \{e\} \notin \mathcal{I}$. Note that Y may have multiple bases, and further, a base of Y may not be a base of a superset of Y. In our example of X and $\mathcal{I}$, in the case of $Y = X$, the bases are $\{b\}$ and $\{a, c\}$ (since $\mathcal{I}$ does not include $\{a, b\}$, $\{b, c\}$, or $\{a, b, c\}$).

We will now use this concept to define a more general notion of independence families parametrized by an integer p, as follows. $(X, \mathcal{I})$ is said to be a *p-system* if for each $Y \subseteq X$, the cardinality of the largest base of Y is at most p times the cardinality of the smallest base of Y:

$$\frac{\max_{J: J \text{ is a base of } Y} |J|}{\min_{J: J \text{ is a base of } Y} |J|} \leq p. \quad (1)$$

To continue with our example of $X, Y, \mathcal{I}$, there are two bases and $p = |\{a, c\}|/|\{b\}| = 2$. For all other choices of $Y \subseteq X$, we have $p = 1$. Thus, $(X, \mathcal{I})$ is a 2-system. The notion of p-systems will be useful in Section 3.6 to prove certain approximation guarantees.

Advantages. Before continuing, it is worth discussing why we are using submodular optimization. A simple alternative would be to just score each event independently, sort the events by score, and return the top events (as long as they fit into the timeline). Such static rankings are insufficient for our purposes since they do not consider diversity. For instance, even though the movie *The Avengers* is very relevant to Robert Downey Jr., the timeline should not solely consist of *The Avengers* events (filming, production, release date, award ceremonies, sequels, *etc.*). Furthermore, we want diversity in the temporal spacing of the events (we show in Section 4 that users strongly prefer diverse timelines) which depends on the time period or zoom factor chosen by the user.

The best way to capture these effects is to reason about the entire *set* of events that should make up the timeline. A submodular set function allows for exactly that, and is able to encourage different notions of diversity, as we show in Section 3.3. In addition, there are computational benefits to using monotone submodular set functions. In particular, as we show in Section 3.6, we can devise a greedy algorithm that enjoys certain optimality guarantees.

3.2 Timeline Optimization Problem

We now formalize our problem. Let $E \subset \mathcal{E}$ be the set of all candidate events for a particular subject entity s constrained to events within the user-specified or default time span. We will display each event as a small box of width w and height h, as shown in Figure 1. Assume we have available screen space of width $\mathcal{W}$ and height $\mathcal{H}$. Let $n = \lfloor \mathcal{H}/h \rfloor$ be the number of boxes that can be stacked vertically within $\mathcal{H}$. Our goal is to find the optimal timeline T^*, which we define as follows:

$$T^* = \arg\max_{T \subseteq E} \text{REL}(s, T) \quad (2)$$
$$\text{s.t. CONSTRAINTS}(T, E, \mathcal{W}, w, n)$$

The relevance function REL evaluates how relevant each event is to the subject and how diverse they are; this is described in more detail in Section 3.3. The temporal constraint CONSTRAINTS requires that all events can fit into the provided space without overlapping or occluding each other; this is described in more detail in Section 3.4.

Note also that our algorithm is able to adapt to different form factors, for example for mobile or desktop, since height and width are just parameters to the optimization algorithm.

3.3 Relevance Function

The function $\text{REL}(s, T)$ captures the quality of the selected subset of events T with respect to the timeline subject s. This is defined as a linear combination of two different kinds of relevance functions:

$$\text{REL}(s,T) = \lambda\, \text{EREL}(s,T) + (1-\lambda)\, \text{DREL}(s,T)$$

where $0 \leq \lambda \leq 1$ trades off the importance of related entities (EREL) versus the importance of related dates (DREL).[2] We define these terms next.

3.3.1 Entity Relevance

We define the relevance of a set of events T to an entity s as follows:

$$\text{EREL}(s,T) = w_1^e\, \text{E2E}(s,T) + w_2^e\, \text{E2EPATH}(T) + w_3^e\, \text{G2E}(T)$$

where E2E measures how relevant the specific *events* are, E2EPATH measures how relevant the *paths* are, and G2E measures how relevant the events are *globally* (*i.e.*, independent of s). As we show shortly, we combine E2E with E2EPATH and G2E to handle data sparsity, *cf.*, backoff-smoothing [19]. This enables inductive reasoning that certain relationships hold generally when we only see a few examples of them. For example, on average, movie roles are more relevant to actors than TV episode roles (E2EPATH). We discuss how we set the weight parameters in Section 4.

In more detail, we measure the entity-to-entity score as follows:

$$\text{E2E}(s,T) = \sum_{re \in \{\text{RE}(e)\ |\ e \in T\}} \text{E2ECOOC}(s, re)$$

where $\text{E2ECOOC}(s, re)$ measures co-occurrence of the entities, and is defined in Section 3.3.3.

Since a path from a subject to a specific entity may occur too rarely to be reliably estimated, we also consider measuring how good the path is, by averaging the co-occurrence score over all entities that can be reached via all the paths in the timeline:

$$\text{E2EPATH}(T) = \sum_{p \in \{\pi_{re}(e)\ |\ e \in T\}} \operatorname*{mean}_{e \in \mathcal{E},\, \pi_{re}(e) = p} \text{E2ECOOC}(\text{SUB}(e), \text{RE}(e))$$

Finally, since even the E2EPATH signal may be too sparse to reliably estimate, we consider G2E, which estimates the global importance of each entity in the timeline:

$$\text{G2E}(T) = \sum_{re \in \{\text{RE}(e)\ |\ e \in T\}} \text{GLOBALIMPORTANCE}(re)$$

We estimate $\text{GLOBALIMPORTANCE}(re)$ as the fraction of search queries that mention the entity re, measured from a 3-month query log (though other measures of global importance could be used instead). Inferring the entity mentioned in a query is done using a proprietary system that applies standard entity linkage algorithms (such as [32]) to the landing page of the query.

[2]We set $\lambda = 0.75$ throughout, based on preliminary experiments on a holdout set showing that users slightly prefer entity relevance to date relevance.

All three functions, E2E, E2EPATH, and G2E, are weighted coverage functions defined over a *set* rather than a *multiset* of related entities (or paths to related entities). As such, they naturally favor content diversity as duplicate entities or paths are only counted once.

3.3.2 Date Relevance

We define the relevance of a set of dates as follows:

$$\text{DREL}(s,T) = w_1^d\, \text{E2D}(s,T) + w_2^d\, \text{E2DPATH}(T)$$

The functions E2D and E2DPATH are defined in a very similar way to their E2E counterparts. For specific dates we have

$$\text{E2D}(s,T) = \sum_{t \in \{\tau(e)\ |\ e \in T\}} \text{E2DCOOC}(s,t)$$

Recall that $\tau(e)$ is the timestamp of event e, so E2DCOOC measures how often an entity and date co-occur, as explained below. Then for the path level we have

$$\text{E2DPATH}(T) = \sum_{p \in \{\pi_t(e)\ |\ e \in T\}} \operatorname*{mean}_{e \in \mathcal{E},\, \pi_t(e) = p} \text{E2DCOOC}(\text{SUB}(e), \tau(e))$$

Similarly, we again use a set instead of a multiset for the timestamps and time paths to favor temporal diversity.

3.3.3 Web-based Co-occurrence Scores

We use co-occurrence signals between entities on the web to capture how related two entities are (E2ECOOC). Similarly, we use co-occurrence between entities and dates (E2DCOOC) to capture how related a particular date is to an entity. We compute these quantities as follows:

1. We run a suite of standard NLP tools (named entity recognition, coreference resolution, *etc.*) over a large corpus of 10B web documents using a set of in-house tools, similar to the Stanford CoreNLP package.[3] We extract entity mentions (which are resolved to Freebase IDs) and date mentions (both year and full date if available), using techniques similar to those described in [32].
2. For each entity mention, we collect all entity-entity and entity-date co-occurrences within a small window around the mention (window of 100 characters or 10-12 words on average).
3. We count these co-occurrences, and convert to probabilities (by normalizing the counts). We define the co-occurrence scores using *normalized pointwise mutual information* (NPMI) as follows:

$$\text{E2ECOOC}(s, re) = \text{NPMI}(s; re) = \frac{\text{PMI}(s; re)}{-\log p(s, re)}$$

$$\text{PMI}(s; re) = \log \frac{p(s, re)}{p(s)\, p(re)}$$

E2DCOOC is defined exactly like E2ECOOC with the only difference that a timestamp t is substituted for entity re.

PMI measures the difference between the co-occurrence probability and the probability expected by chance if the events were independent. It is critical to account for particularly popular entities (*e.g.*, Barack Obama) or dates (*e.g.*, 2014), and dividing the co-occurrence probability by the popularity of the co-occurring entities/dates is a principled way of achieving this.

NPMI normalizes PMI to the range $[-1, 1]$. Since we are only interested in the most related pairs of entities (or entity/date pairs), we only retain positive NPMI scores. Furthermore, we require that this co-occurrence was extracted from at least five different web

[3] http://nlp.stanford.edu/software/corenlp.shtml

domains for robustness. Computing co-occurrence statistics from a large web corpus (10B documents) and generating candidate events (for over 1M entities) takes about six hours using map-reduce.

3.3.4 Submodularity of Objective Function

We now show that our objective function is a monotone submodular set function.

Theorem 3.1. *Let $f(T) = \text{REL}(s, T)$ for any given subject s. Then $f : 2^T \to \mathbb{R}^+$ is a monotone submodular set function.*

Proof. First we note that f is a non-negative linear combination of weighted coverage functions (since $w_1^e, w_2^e, w_3^e, w_1^d, w_2^d \geq 0$). Each of these are known to be submodular [14, 20], which is easy to see, due to the diminishing returns property of weighted coverage functions. Furthermore, a non-negative linear combination of submodular functions is submodular as well.

Second, we note that f is also monotone. This holds since all individual weighted coverage functions are non-negative (because E2ECooc, E2DCooc, and GlobalImportance are all non-negative), so adding up more terms makes the sum bigger.

Third, we have $f(\emptyset) = 0$, again because all weighted coverage functions lead to empty sums. □

3.4 Temporal Diversity Constraint

The layout constraint requires that all events can fit into a timeline of width $\mathcal{W}$ and height $\mathcal{H}$ without overlap. We consider a simple layout strategy: if the boxes (of width w) depicting two events have a temporal overlap, we can stack one on top of the other, as shown in Figure 1, but we require that the height of this stack be at most $n = \lfloor \mathcal{H}/h \rfloor$.

We can define this constraint more formally as follows. Recall that each event $e \in E$ has a timestamp $\tau(e) \in \mathbb{R}$. Let R be an interval $R = [a, b] \subset \mathbb{R}$. We denote the set of events in $T \subseteq E$ with timestamps within R as $T \sqcap R = \{e \in T \mid \tau(e) \in R\}$. We define t_w as the length of a time period that corresponds to the width of w on the timeline; t_w can be easily computed according to $\mathcal{W}, w$, and the beginning and ending timestamps for E. Finally, we say a set $T \subseteq E$ of events satisfies the layout constraint $\text{CONSTRAINTS}(T, E, \mathcal{W}, w, n)$ if

$$\forall t \in \mathbb{R} : |T \sqcap [t, t + t_w)| \leq n \tag{3}$$

This constraint can be interpreted as follows: for any point in time $t \in \mathbb{R}$, draw a line of width up to t_w starting at t. Consider the intersection of all timestamps in T with this line. If the size of the intersection is less or equal to n, then we know that we can vertically stack the events in the intersection without violating the height constraint.

It turns out that this constraint forms a p-system that enables us to prove approximation guarantees (see Section 3.6).

Theorem 3.2. *Let $(E, \mathcal{I})$ be an independence family based on our layout constraint where $T \in \mathcal{I}$ if $T \subseteq E$ satisfies Equation* (3)*. Then $(E, \mathcal{I})$ forms a p-system for $p = 2$.*

Proof Sketch. For the full proof please refer to the appendix of the full version of this paper [3]. The idea is as follows: We need to show that $|J_{\max}|/|J_{\min}| \leq 2$ where $J_{\min}$ and $J_{\max}$ are minimal and maximal bases of an arbitrary subset $T \subseteq E$. We can show that $J_{\max}$ can be at most twice as large as $J_{\min}$ by "deleting" all elements from $J_{\max}$ eventually, where in each step we delete an element $b \in J_{\min}$, and up to two elements in $J_{\max}$ (if they exist) in close proximity to b. Intuitively this process works because we can never have an element in $J_{\max}$ that we cannot delete in this way since then either $J_{\min}$ has too few points to be a maximal independent set (base), or $J_{\max}$ has too many points to be an independent set (it would violate the layout constraint). Both would contradict our assumption that both $J_{\min}$ and $J_{\max}$ are bases of Y. □

Algorithm 1 GreedyTimeline

1: $\hat{T} \leftarrow \emptyset, C \leftarrow \emptyset$
2: **repeat**
3: $\quad C \leftarrow \{e \in E \setminus \hat{T} \mid \hat{T} \cup \{e\} \in \mathcal{I}\}$
4: $\quad$ **if** $C \neq \emptyset$ **then**
5: $\quad\quad e \leftarrow \arg\max_{e' \in C} \text{REL}_{\hat{T}}(e')$
6: $\quad\quad \hat{T} \leftarrow \hat{T} \cup \{e\}$
7: **until** $C = \emptyset$
8: **return** $\hat{T}$

3.5 Optimization Algorithm

Our problem is reduced to the problem of finding a solution T^* that obtains $\max_{T \in \mathcal{I}} \text{REL}(s, T)$, where $\mathcal{I}$ is the independence family that we defined in Section 3.4 (see Theorem 3.2). Unfortunately, such problems are NP-hard for many classes of submodular functions, including weighted coverage [14] (our case). Therefore we focus our attention on efficient algorithms with theoretical approximation guarantees.

As we show in Section 3.6, a greedy algorithm (see Algorithm 1) has certain approximation guarantees. The algorithm incrementally builds an approximate solution $\hat{T}$ (without backtracking), starting with the empty set. In each iteration it adds an element e from the set of valid candidates C that most improves the current solution (according to the marginal gain $\text{REL}_{\hat{T}}(e)$), while maintaining independence of the solution (see line 3).

This greedy algorithm has complexity $\mathcal{O}(|E|^2)$ (assuming computing $\text{REL}_{\hat{T}}(\cdot)$ takes constant time). In practice, it can be sped up significantly in practice by using lazy evaluations, as first proposed in [27] (see also [21] and Section 2 of [20]). This "lazy greedy" algorithm exploits the fact that the marginal gain for each element only decreases with each iteration; that is, $\text{REL}_{\hat{T}}(e) \geq \text{REL}_{\hat{T}'}(e)$ for $\hat{T} \subseteq \hat{T}'$, and therefore we can use previously computed values as upper bounds to save many evaluations of $\text{REL}_{\hat{T}}$. We use this more efficient implementation of the greedy algorithm.

3.6 Approximation Guarantee

Before we can prove our main theoretical result, we introduce the following lemma. It is proved in [28] for the special case where $\mathcal{I}$ is defined by the intersection of p matroids on X, and for the more general case of p-systems in Appendix B of [7].

Lemma 3.3. *[7, 28] The algorithm* GreedyTimeline *to compute $\max_{S \in \mathcal{I}} f(S)$, where $(X, \mathcal{I})$ is a p-system and $f : 2^X \to \mathbb{R}^+$ is a monotone submodular set function, has a tight approximation ratio of $1/(p + 1)$.*

We have shown in Theorem 3.1 that REL is a monotone submodular set function and in Theorem 3.2 that the temporal constraints CONSTRAINTS form a p-system for $p = 2$. Following Lemma 3.3, our greedy algorithm has the following approximation bound.

Theorem 3.4. *Algorithm* GreedyTimeline *has an approximation ratio of $1/3$; that is,*

$$\text{REL}(s, \hat{T})/\text{REL}(s, T^*) \geq 1/3,$$

for any subject s, where $\hat{T}$ is the output of our algorithm GreedyTimeline*, and T^* is the optimal solution.*

3.7 Zooming in or out of the Timeline

We have proposed an efficient algorithm for generating timelines. This efficiency (in particular, the "lazy greedy" property

that allows us to re-use previously computed values) enables us to quickly (re)compute the optimal timeline if the user chooses to dynamically zoom in or out of a specific time period. In practice, we observe running times roughly linear in the number of events taking a few hundred milliseconds which is much faster than the quadratic theoretical worst case bound. An example was given in Figure 4, where we show the timeline for the most recent few years of Robert Downey Jr.'s life (*cf.*, Figure 1).

The default interval for each timeline is computed as follows: we choose the shortest time period that covers at least 90% of all generated events (restricted to the lifetime of the entity). Note that for person entities, this time period usually corresponds to less than 90% of their lifetime based on the intuition that most interesting events happen to people after they grow up, but before they retire. (See Section 6 for a discussion of when this heuristic can fail.)

4. EVALUATION

In this section, we evaluate the quality of our method for producing timelines[4]. Since there is no ground truth to compare to, we asked Amazon Mechanical Turk raters to vote for their preferred timeline. We do this in a series of paired comparisons in which we vary one component of the algorithm at a time, resulting in six different models[5], summarized in Table 1. The results are shown in Table 2 and Figure 6, and are explained shortly.

In summary, our experiments show that (1) users always significantly prefer our full method over baseline and state-of-the-art methods; (2) enforcing temporal diversity and content diversity significantly improves the results; and (3) both entity relevance and date relevance contribute to generating a quality entity timeline.

4.1 Experimental setup

We generated timelines for 250 popular entities (75 music artists and bands, 75 actors, 50 politicians, 50 athletes) for each of the six methods in Table 1. We chose popular entities instead of random or tail entities because evaluations cannot be trusted on entities that most raters are not at all familiar with. Popular entities also account for the major share of the total query volume and their large number of candidate events and often long lifespan makes timeline generation particularly challenging. Furthermore, we chose to evaluate timelines through pairwise preferences rather than absolute quality judgments as this has often been found to be less subjective and thus more reliable [9].

Let $T(e, m)$ denote the timeline for entity e generated by model m; let $m = 0$ denote the full model (the control), and let $m = 1 : 5$ denote one of the ablated models (experimental conditions; described in Table 1 and the following sections). For each entity e, we displayed the control timeline $T(e, 0)$ and the experimental timeline $T(e, m)$ for $m > 0$, one above the other; we randomized the decision whether the experiment or control was shown on top.

We asked each rater which timeline they preferred, on a 5-point scale, corresponding to strongly preferring the top one, slightly preferring the top one, being neutral, slightly preferring the bottom one, and strongly preferring the bottom one. We also asked each rater to give qualitative comments to justify their decision, to gain further insight. Each pair of timelines is rated by five different raters (1154 distinct raters in total). We encouraged raters to research each entity (*e.g.*, using Wikipedia) before evaluating each timeline about that entity; fortunately, 79% of raters reported that they were already familiar with these entities.

[4]A demo is available at `http://cs.stanford.edu/~althoff/timemachine`

[5]Experiments on varying w_1, w_2, w_3 show that the results are insensitive to the exact parameter values as long as $w_1 \gg w_2 \gg w_3$ (*cf.*, backoff-smoothing [19], see Section 3.3.1).

Name	w_1^e	w_2^e	w_3^e	w_1^d	w_2^d	TD	CD
FULL	1	10^{-2}	10^{-4}	1	10^{-2}	1	1
BASE	0	0	1	0	0	1	1
FULL-E2D	1	10^{-2}	10^{-4}	0	0	1	1
FULL-E2E	0	0	10^{-4}	1	10^{-2}	1	1
FULL-TD	1	10^{-2}	10^{-4}	1	10^{-2}	0	1
FULL-CD	1	10^{-2}	10^{-4}	1	10^{-2}	1	0

Table 1: Summary of experimental configurations. We fix $\lambda = 0.75$ throughout. TD = temporal diversity, CD = content diversity. FULL-TD means the full model without temporal diversity, etc. Note that all methods remove duplicate events, which is a minimal form of content diversity, but if CD=1, we ensure diversity amongst types of events (entities and paths) as well; see Section 4.5 for details.

Ablated model	#Tasks	#Raters	RAggr	RPref
BASE	1250	344	77.0%	83.8% ***
FULL-E2D	1250	463	75.7%	59.8% **
FULL-E2E	1250	676	73.2%	64.3% ***
FULL-TD	150	53	75.3%	86.7% ***
FULL-CD	1250	665	81.0%	91.1% ***

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Table 2: Summary of the user studies. Each row shows an ablated version that was compared to the full model. The asterisks represent the p-value corresponding to a Binomial hypothesis test that compares the RPref value to 50%.

To simplify the analysis, we collapsed the user votes to a 3-point scale: prefer control (full model), neutral, or prefer experiment (ablated model). Let $V(e, m, r) \in \{F, T, A\}$ be the vote by rater $r \in R$ for entity $e \in E$ and method $m \in M$, where F represents preferring the full model, T represents a tie, and A represents preferring the ablated model. Let $N(e, m, v) = |\{r \in R : V(e, m, r) = v\}|$ be the number of raters who voted for category $v \in \{F, T, A\}$, for entity e, and method m. Let $M(e, m) = \arg\max_v N(e, m, v)$ be the majority vote.

We compute *agreement* between raters (RAggr) as the fraction of raters agreeing with the majority vote (including tie votes and tied majorities):

$$RAggr(m) = \frac{|\{V(e, m, r) = M(e, m) : e \in E, r \in R\}|}{|\{V(e, m, r) : e \in E, r \in R\}|}$$

We define the rater *preference* (RPref) for the full method as the fraction of times the majority of raters vote for the full method, excluding cases where there is no clear majority; that is, we set $M(e, m) = \text{NULL}$ if the majority is not unique (*e.g.*, if we have 2 votes for F, 2 for A, and 1 for T):

$$RPref(m) = \frac{|\{M(e, m) = F : e \in E\}|}{|\{M(e, m) \in \{F, A\} : e \in E\}|}$$

(Note that a 5:0 vote in favor of full (5 F, 0 A) is treated the same as a 3:2 vote.) If both methods are equally good, we would expect both the full and the ablated model to win exactly 50% of the time; that is, $RPref(m) = 0.5$ (our *null hypothesis*). This allows us to use a simple two-sided Binomial hypothesis test of significance.

4.2 Baseline Algorithms

In our initial trial, we defined the baseline algorithm as follows: rank all the candidate events by the G2E global entity score, and then show the top K events (where each event is represented by a box on a timeline of width 1000 pixels, and we allow up to

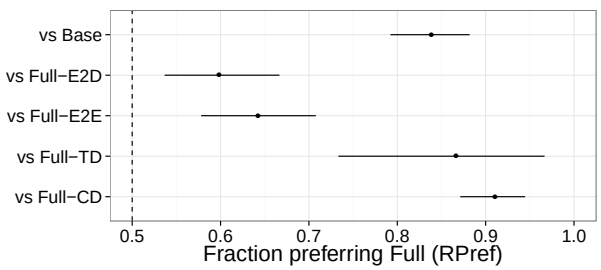

Figure 6: Fraction of entities for which raters preferred Full approach over ablated version (RPref) along with bootstrapped 95% confidence intervals. Results show significant preference for our proposed approach in all cases.

$n = 2$ boxes to be stacked vertically). However, we found that the G2E score sometimes picked the same related entity more than once (*e.g.*, if Robert Downey Jr. starred in *Iron Man* and later won an award for it, we may display *Iron Man* twice, at two different time points). Users strongly disliked this in preliminary experiments, so we decided to augment the baseline through post-filtering all results by removing duplicate entities from all methods (except Section 4.5), keeping the highest scoring event in each case (we never allow duplicate events).

In addition, we noticed that the baseline model would sometimes result in visual crowding, since it does not enforce temporal diversity. Users strongly disliked this as well, so we decided to further improve the baseline by post-processing all results and removing temporally overlapping events, keeping the highest scoring event in each case.

In practice, we can implement this modified baseline BASE by using our constrained submodular optimization algorithm, but setting the weights so that $w_1^e = w_2^e = w_1^d = w_2^d = 0$ and $w_3^e = 1$, thus putting all the emphasis on the G2E signal. This is because the greedy optimization algorithm will ensure that it never adds an event that temporally overlaps an existing event. Most of the time our submodular coverage function does not yield any duplicate entities. In the rare case that the optimization does lead to duplicate entities, we explicitly remove them to ensure that entity diversity has no impact on any other experiment (except Section 4.5 that explicitly measures the importance of content diversity).

We can see from the results (Table 2 and Figure 6) that on average, 84% of the time raters prefer our full model (significant difference at $p < 0.001$ according to a Binomial test). This shows that a global relevance score is inadequate, even when augmented by temporal diversity and content diversity.

The only other relevant baseline known to us is the CATE system described in [39]. CATE ranks related entities by co-occurrence with the timeline entity within documents of a given context. This approach is very similar to the E2E approach described in Section 3.3.1. We consider our E2E signal as an improvement over the CATE baseline since, first, we only consider more direct connections between entities in a KB compared to co-occurrence within the same document. Second, our relevance signal captures co-occurrence on a large web corpus within a small window which gives higher coverage and more focus compared to document-wide co-occurrence within Wikipedia only. And third, we perform subset selection instead of a static ranking which allows selected events to influence which other events are selected next. Section 4.3 shows that our methods outperform E2E.

4.3 Evaluating Relevance Signals

To compare the different ways of measuring event relevance, we performed two experiments. First, we "turned off" the date signal DREL, by setting $w_1^d = w_2^d = 0$. We call this model Full-E2D, meaning the full model without the E2D signal. Raters prefer our full model to this version about 60% of the time, which is a significant difference at the level of $p < 0.01$ (Binomial test).

The utility of the date signal depends on which kind of entity we are creating a timeline for. For people, it is common to find the date of birth, death, marriage or other key events to be explicitly mentioned on the web; this makes it relatively easy to determine that these events are important.[6]

Second, we "turned off" the EREL signal, by setting $w_1^e = w_2^e = 0$. We call this model Full-E2E. Raters prefer our full model to this about 64% of the time, indicating that the E2E signal is somewhat more important than E2D (significant at $p < 0.001$).

However, the benefit of the E2E signal varied by domain/vertical: we found it most useful for actors and athletes, whereas for musicians, the E2D signal was more helpful. We attribute this to conventions in what entities and dates are co-mentioned on the web (in close proximity). E2D works well for music artists because important dates such as album release dates and tour dates are frequently mentioned across many websites (online stores, ticketing websites, etc.). However, this is different in the movie domain. There are many more entities related to the movie (director, producer, dozens of actors, etc.) and only a few of them will be highlighted in close proximity to the movie release date (usually one or two star actors). How helpful the E2D signal is depends on what usually gets mentioned in close proximity of the date, which is subject to certain conventions and marketing decisions. For instance, the first *Pirates of the Caribbean* movie (2003) has a lower E2D score for actor Johnny Depp than later sequels even though the first movie was the bigger milestone for Johnny Depp's career. The sequel promotions just featured Johnny Depp (who had gained in popularity) more prominently. We found the E2E signal to be more generally applicable and less influenced by such effects.

We further found that the E2D signal has less utility when events do not exhibit a clear temporal focus such as long military conflicts (compared to birth/death/marriage dates or concert tours). In these cases, the E2E signal is helpful in providing additional information in the event selection phase.

4.4 Evaluating Temporal Diversity

As we mentioned in Section 4.2, users strongly dislike when displayed events overlap in time, since it is not easy to see the corresponding images and descriptions. Indeed, we see that in 86% of the experiments, raters prefer our full model over an ablated version, which we call Full-TD, that maximizes relevance and content diversity but without any temporal constraint (significant at $p < 0.001$). This is despite the fact that the ablated model also includes the simple overlap filter we described in Section 4.2. The number of events we show is controlled and set to the number of events in our Full approach, as we aim to measure the impact of temporal diversity while controlling for the amount of information shown (though the overlap filter may remove some of them). The reason the full model works better is that it can take into account the temporal overlap during the optimization process, so if one event is removed, another (non-overlapping) event can be added instead.

4.5 Evaluating Content Diversity

As we mentioned in Section 4.2, users strongly dislike when the same entity is repeated (with different timestamps), so we always remove such cases from all methods. Here, we quantify the impor-

[6]Of course, our system treats birth and death dates as special, since they inform the beginning and end of the timeline for a person (see Section 3.7).

tance of content diversity in generating timelines. Note that in addition to entity diversity there are other, slightly more subtle forms of content diversity that we might wish to consider. For example, we might not want to list only the different movies that an actor has been in, even if they all have high relevance scores; instead we wish to include awards, TV shows, and personal relationships as well. Our submodular set cover objective captures this by using the E2EPATH feature, which gives higher score to a set of events with distinct path types (see Section 3.3.1).

To evaluate this, we consider an alternative model in which we evaluate the score by summing over the multiset (rather than set) of related entities (or paths to related entities), allowing for duplicate entities or paths during the optimization process; we call this Full-CD. We see that raters prefer our full model 91% of the time compared to this ablated model (significant at $p < 0.001$). Again, we attribute this to the fact that the full model is aware of the penalty for duplication during the optimization process, and can adjust its output appropriately.

5. RELATED WORK

There has been much work on extracting temporal events from text [17, 25], and in summarizing large text corpora such as tag streams [13], news corpora [11], Wikipedia biographies [5], and Wikipedia edit histories [38]. There has also been work on mining temporal patterns across such textual data sets [16, 36, 41].

Another body of related work concerns document summarization [2, 34]. The evaluation of summarization approaches has always been challenging, and measures like Rouge [23] are often used if ground truth summaries are available. In our case, we use paired comparisons, since we do not have ground truth.

The summarization and IR communities have identified diversity as an important quality criterion [8, 24]. More recently, research has focused on complementing traditional corpus-based relevance measures with signals such as social attention [42]. Early work on timeline generation by Swan and Allan [37] attempted to summarize a news corpus by displaying major events and topics along a timeline. In a similar spirit, Shahaf et al. [31] have created maps of information that summarize complex storylines across news documents. Similar techniques have been applied to scientific literature [30, 33]. Our paper extends this line of work by using multiple relevance signals (based on web co-occurrence), as well as showing that content and temporal diversity are critical for quality timelines.

Submodular optimization has been shown to be a powerful framework for summarization [18, 24, 30, 31, 33], since it naturally captures notions of diversity through its diminishing returns properties [10]. Furthermore, there are efficient approximation algorithms with theoretical guarantees [1, 7, 20, 21, 27, 28].

Some recent work has focused on generating personalized timelines based on Facebook [15] or Twitter feeds [22]. Timelines generated based on information from KBs have been considered in [26, 39, 40]; these papers are the ones most related to our approach. However, there are several differences. First, [26, 40] do not consider a ranking of individual events (required when space is limited) nor visual space constraints, so there is no optimization algorithm involved. Instead, they simply display all events which is not an option in our context as each timeline entity might have hundreds of candidate events (see Figure 5). We have empirically shown that it is absolutely necessary to address relevance, redundancy, and space constraints to generate quality timelines. Second, [39] considers ranking related entities but uses a different notion of relatedness (sharing many contexts rather than more direct connections in the KB). In this approach, it is impossible to capture relationships between selected events as the ranking is static. To the best of our knowledge, this is the only relevant baseline and we show in Section 4.3 that our proposed method outperforms an improved reimplementation of this approach. Third, none of these papers conducts any quantitative evaluation of their timelines.

Finally, we should mention that Bing [29] has released a system called "Timeline" that is somewhat similar to ours. However, there are (to the best of our knowledge) no published accounts of how their system works. Furthermore, their timelines are static, and do not allow the user to interact with the timeline, a feature which we consider to be very important, especially for mobile browsing.

6. FUTURE WORK

In this section, we suggest some directions for future work, based in part on the comments written by the raters.

Choosing a better default timespan. As we discussed in Section 3.7, the algorithm picks a default time span for a person that covers 90% of their generated life events. However, sometimes this is suboptimal. For example, consider the US president John F Kennedy: many important events occurred in the last few years of his life (assassination, presidency, Cuban missile crisis, Bay of Pigs invasion, *etc.*). Our default timespan misses many of these. In particular, his assassination, his presidency, and his marriage to Jacqueline Kennedy Onassis are chosen first, and these then block other important events such as Cuban Missile Crisis or his involvement in the Vietnam war.

We address this problem by allowing the user to zoom in to the appropriate period. Other potential solutions include using the E2D scores to weight some time periods higher than others, and including the search over suitable time periods as part of the optimization.

Time points vs intervals. Our algorithm represents events based on a single point in time. However, some events (*e.g.*, wars) are more naturally associated with intervals. Currently our method may pick the start or end of a war, but might not show both, due to the diminishing returns property. This could be fixed by modifying the algorithm to reason about temporal intervals.

Choosing how to describe an event. Sometimes a related entity is connected to the subject via many different paths, and all have the same timestamp. In this case, it is hard to know which relation to show to the user. For example, the system sometimes describes a date associated with someone's death as the end of their marriage; while technically true, this is rather unintuitive. Another example concerns US presidents: sometimes such people are described as being a military commander. Again, while technically true (since the US president is also the Chief of the Armed Forces), this is unintuitive to users. We may be able to fix this problem by learning a ranking model applied to particular candidate values for any given subject and relation or by influencing the way the data is curated.

User preferences and subjectivity. In some cases, raters did not agree on which timeline was best. The reason often seems to boil down to individual preferences. In our experiments, the biggest area of disagreement is over how much the timeline should be focused on professional life (*e.g.*, jobs, albums, books) vs personal life and relationships (*e.g.*, marriage, children). Users had different opinions, even for the exact same timeline subjects, which illustrates the need for personalization in this space. One approach would be to distinguish between professional and personal events, and to allow some trade-off parameter between them.

Extractive vs abstractive summarization. Our current approach to building a timeline is similar to "extractive summarization" techniques in the NLP community, in the sense that we select a set of events from a candidate pool. However, sometimes this is suboptimal, since the relationship between two entities may be more complex. For example, Robert Downey Jr.'s father (Robert Downey Sr.)

shows up on his timeline, but is described being a co-star in a movie rather than being his father. While technically correct, it would be more satisfying to create an abstract summarization of the relationship, describing that Robert Downey Sr. is both the father and a co-star. We leave this to future work.

Creating timelines for collections. In the future, we would like to go beyond timelines for single entities, and derive a method to summarize collections of entities (*e.g.*, 1930s jazz artists), periods of time (1920s in the U.S.), or long-lasting events (World War II).

7. CONCLUSIONS

We presented a system called TIMEMACHINE for automatic timeline generation for entities in a knowledge base. The timeline generation problem is formulated in a submodular optimization framework that jointly optimizes for relevance, content diversity and temporal diversity. Web-based co-occurrence signals are used to determine the relevance of other entities and dates to the timeline subject. We proved that an efficient greedy approximation algorithm achieves near-optimal performance. The proposed approach is evaluated through a comprehensive series of user studies demonstrating that both temporal diversity and content diversity are crucial, and that web-based co-occurrence signals significantly improve over a baseline model that relies on global importance.

Acknowledgments. We thank Evgeniy Gabrilovich for many helpful discussions, Arun Chaganty, Stefanie Jegelka, Karthik Raman, Sujith Ravi, and Ravi Kumar for their insights on submodular optimization, Jeff Tamer and Patri Friedman for their support with the user studies, Danila Sinopalnikov and Alexander Lyashuk for their help with the co-occurrence pipeline, and Jure Leskovec, David Hallac, Caroline Suen, and the anonymous reviewers for their valuable feedback.

8. REFERENCES

[1] A. Ahmed, C. H. Teo, S. Vishwanathan, and A. Smola. Fair and balanced: Learning to present news stories. In *WSDM*, 2012.

[2] J. Allan, R. Gupta, and V. Khandelwal. Temporal summaries of new topics. In *SIGIR*, 2001.

[3] T. Althoff, X. L. Dong, K. Murphy, S. Alai, V. Dang, and W. Zhang. TimeMachine: Timeline Generation for Knowledge-Base Entities. *arXiv:1502.04662*, 2015.

[4] R. A. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, New York, 1999.

[5] D. Bamman and N. Smith. Unsupervised discovery of biographical structure from text. *TACL*, 2(10):363–376, 2014.

[6] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 2008.

[7] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[8] J. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, 1998.

[9] B. Carterette, P. N. Bennett, D. M. Chickering, and S. T. Dumais. Here or there: Preference Judgments for Relevance. In *Advances in Information Retrieval*. Springer, 2008.

[10] A. Dasgupta, R. Kumar, and S. Ravi. Summarization through submodularity and dispersion. In *ACL*, 2013.

[11] Q. X. Do, W. Lu, and D. Roth. Joint inference for event timeline construction. In *EMNLP-CoNLL*, 2012.

[12] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *SIGKDD*, 2014.

[13] M. Dubinko, R. Kumar, J. Magnani, J. Novak, P. Raghavan, and A. Tomkins. Visualizing tags over time. *TWEB*, 1(2):7, 2007.

[14] U. Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[15] D. Graus, M.-H. Peetz, D. Odijk, O. de Rooij, and M. de Rijke. yourHistory–Semantic linking for a personalized timeline of historic events. *Workshop: LinkedUp Challenge at OKCon*, 2013.

[16] T. Huet, J. Biega, and F. M. Suchanek. Mining history with le monde. In *AKBC*, 2013.

[17] H. Ji, T. Cassidy, Q. Li, and S. Tamang. Tackling representation, annotation and classification challenges for temporal knowledge base population. *KAIS*, 2013.

[18] A. Kannan, S. Baker, K. Ramnath, J. Fiss, D. Lin, L. Vanderwende, R. Ansary, A. Kapoor, Q. Ke, M. Uyttendaele, et al. Mining text snippets for images on the web. In *SIGKDD*, 2014.

[19] S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. In *IEEE Trans Sig. Process.*, 1987.

[20] A. Krause and D. Golovin. Submodular function maximization. In *Tractability: Practical Approaches to Hard Problems (to appear)*. Cambridge University Press, 2014.

[21] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *SIGKDD*, 2007.

[22] J. Li and C. Cardie. Timeline generation: tracking individuals on twitter. In *WWW*, 2014.

[23] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Proc. ACL Text Summarization Workshop*, 2004.

[24] H. Lin and J. A. Bilmes. Learning mixtures of submodular shells with application to document summarization. In *UAI*, 2012.

[25] X. Ling and D. S. Weld. Temporal information extraction. In *AAAI Conference on Artificial Intelligence*, 2010.

[26] A. Mazeika, T. Tylenda, and G. Weikum. Entity timelines: Visual analytics and named entity evolution. In *CIKM*, 2011.

[27] M. Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*, pages 234–243. Springer, 1978.

[28] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions — I. *Mathematical Programming*, 14(1):265–294, 1978.

[29] R. Qian. Timeline: Understanding Important Events in People's Lives. `http://blogs.bing.com/search/2014/02/21/timeline-understanding-important-events-in-peoples-lives/`, February 2014. Last retrieved on Feb 18, 2015.

[30] D. Shahaf, C. Guestrin, and E. Horvitz. Metro maps of science. In *SIGKDD*, 2012.

[31] D. Shahaf, J. Yang, C. Suen, J. Jacobs, H. Wang, and J. Leskovec. Information cartography: creating zoomable, large-scale maps of information. In *SIGKDD*, 2013.

[32] W. Shen, J. Wang, and J. Han. Entity linking with a knowledge base: Issues, techniques, and solutions. *TKDE*, 2015.

[33] R. Sipos, A. Swaminathan, P. Shivaswamy, and T. Joachims. Temporal corpus summarization using submodular word coverage. In *CIKM*, 2012.

[34] K. Spärck Jones. Automatic summarising: The state of the art. *Information Processing & Management*, 43(6):1449–1481, 2007.

[35] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *WWW*, 2007.

[36] F. M. Suchanek and N. Preda. Semantic Culturomics (Vision paper). In *Very Large Databases (VLDB)*, 2014.

[37] R. Swan and J. Allan. Automatic generation of overview timelines. In *SIGIR*, 2000.

[38] T. Tran, A. Ceroni, M. Georgescu, K. D. Naini, and M. Fisichella. Wikipevent: Leveraging wikipedia edit history for event detection. In *WISE*. Springer, 2014.

[39] T. A. Tuan, S. Elbassuoni, N. Preda, and G. Weikum. CATE: Context-Aware Timeline for Entity Illustration. In *WWW*, 2011.

[40] Y. Wang, M. Zhu, L. Qu, M. Spaniol, and G. Weikum. Timely Yago: harvesting, querying, and visualizing temporal knowledge from Wikipedia. In *EDBT*, 2010.

[41] G. Weikum, N. Ntarmos, M. Spaniol, P. Triantafillou, A. A. Benczúr, S. Kirkpatrick, P. Rigaux, and M. Williamson. Longitudinal Analytics on Web Archive Data: It's About Time! In *CIDR*, 2011.

[42] X. W. Zhao, Y. Guo, R. Yan, Y. He, and X. Li. Timeline generation with social attention. In *SIGIR*, 2013.