Linea: Building Timelines from Unstructured Text

Tiago Etiene
Modelo Inc., San Francisco, USA

Paulo Pagliosa
FACOM–UFMS, Campo Grande, Brazil

Luis Gustavo Nonato
ICMC–USP, São Carlos, Brazil

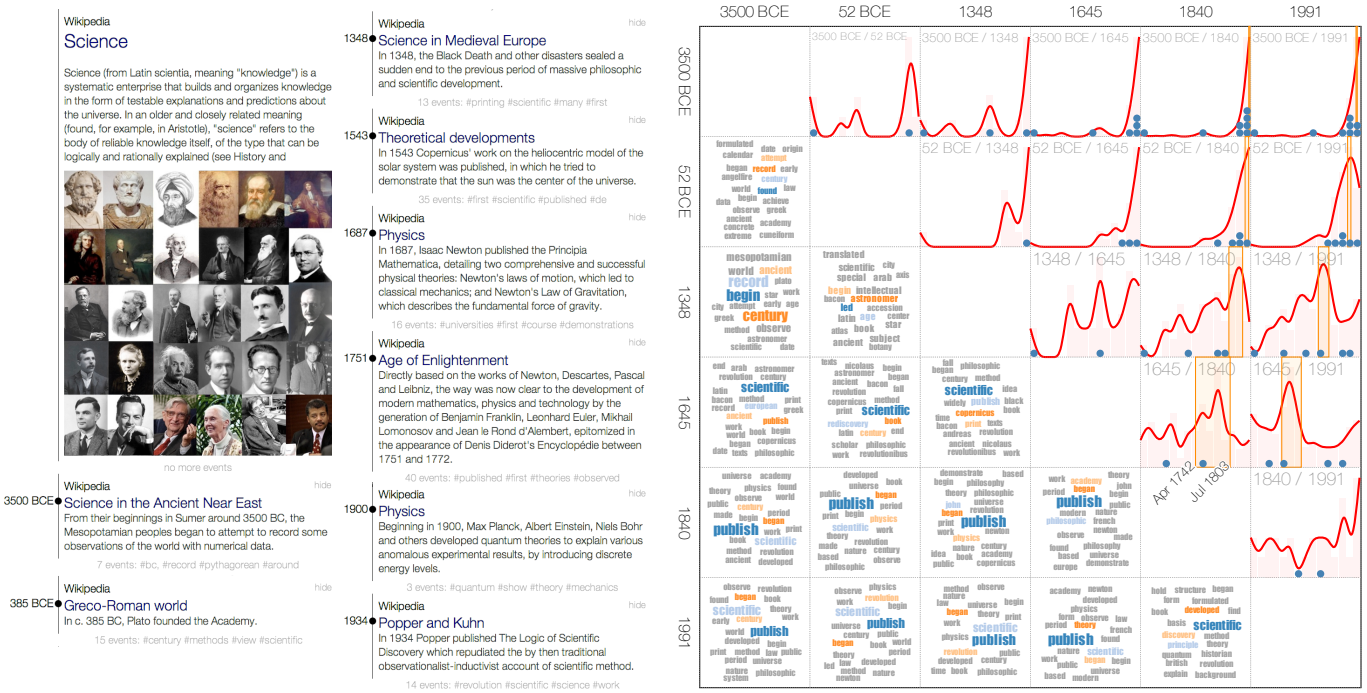


Fig. 1. A custom-made timeline of the history of science (left) and its event matrix (right). Each cell in the upper triangular part of the event matrix contains a curve depicting the distribution of events of the input data in that time interval. Peaks represent important periods of time. The timeline on the left shows the user-selected events. Blue dots represent the events shown on the timeline.

***Abstract*—Timelines are often used for summarizing complex, time-evolving, sequences of events. In this work, we propose *Linea*, a tool that helps users build timelines from unstructured text. Besides providing interactive tools for browsing, selecting, editing, and filtering events, *Linea* includes mechanisms to build smart defaults, thus providing good starting points for users to create their own timelines. The core component is a novel visualization widget called *event matrix*, a matrix designed to explore events over time and in different time scales. We illustrate the strengths of our tool with a case study showing that important events can be browsed, timelines can be easily built, and show the result of an information user evaluation.**

***Keywords*-timeline; textual summaries; event matrix.**

I. Introduction

Building meaningful summaries from document collections is a routine task for many professionals. As an example, when preparing a news article, journalists gather as much information as possible about the topic, select events of interest, and organize the flow of events. In the era of information overload, semi-automated techniques for text analysis become indispensable to explore the document haystack. Nevertheless, although computational tools for assisting the summarization and visualization of time-stamped documents are available, the burden of compiling, organizing, and explaining events of interest still falls on the shoulders of professionals.

In this paper, we focus on the summarization of unstructured text via timelines. Timelines can be useful in the classroom [1], for journalist [2], in museums, and others. Given a set of unstructured text data, our challenges are (i) to extract real world events from the text, (ii) show those events to the user, and (iii) provide tools that allows the user to explore, filter, and build timelines based on the extracted events. Our approach, called *Linea*, combines text analysis techniques used by the Information Retrieval and Natural Language Processing communities to extract events, with a novel visualization widget, the *event matrix* to allow users to explore them. Our main contributions are:

- *Event matrix*, a visualization widget designed to explore events over time and in different time scales;
- *Linea*, a tool that combines text analysis techniques, visualization, and interaction tools for building timeline summaries.

II. Related Work

Here we focus on techniques and systems that rely on timelines to convey information from time-stamped data. A comprehensive discussion can be found in the surveys by Wolgang and Heidrun [3] and Srivatsan and Shanti [4].

A. Visualization

The use of timelines as a visual resource to convey the content of data endowed with temporal information dates back from eighteenth century [5]. Recently, computational systems have been proposed to handle large amounts of data, most of them influenced by the gripping ThemeRiver methodology [6], [7]. ThemeRiver relies on stacked graphs to build meaningful visualizations where horizontal axis reflects flow of time and vertical dimension is segmented in strips (or flows), each one representing a theme. The width of the strips changes according to the strength of the underlying theme in each time slice. ThemeRiver metaphor has been improved and adapted in many ways. For instance, TextFlow [8] combines the river metaphor with sophisticated mechanisms to detect birth, death and topic splitting, making use of glyphs and polylines to highlight topic dynamics as well as their interactions. TIARA [9] builds upon ThemeRiver by incorporating automatic text summarization techniques to assist users in comprehending and analyzing abstract and complex text summaries. EventRiver [10] relies on river metaphor, but strips are replaced by geometric entities representing bubbles whose dimensions reflect the importance and duration of each event.

Tanahashi and Ma [11] and Liu et al. [12] employ thread-based timelines for generating storyline visualizations. Thread-based layouts have also been employed to visualize document collections [13], where elaborate mechanisms that identify the dynamic competition among topics have been proposed [14].

Systems such as LifeFlow [15], LifeLine [16], OutFlow [17], PatternFinder [18], and TimeLine [19] employ timelines as a facet of multifaceted visualization systems to clearly convey patient data histories. These systems are typically made up of advanced interfaces able to promptly depict images, medical records, and temporal patterns contained in medical histories. SemaTime [20] relies on a multifaceted system that enables hierarchical categorization and visualization of time-dependent entities in document collections. Continuum [21] is a faceted system where a timeline enriched with event histogram is used to provide an overview of the whole content of a data set. A detailed view panel is used to show users selected information. The level of detail is controlled by users through interactive slide bars.

B. Summarization

Several techniques are available for automatic text summarization. LexPageRank [22] assigns importance to document sentences by using the PageRank algorithm. Strötgen et al. [23] introduces a new search method that takes into account temporal information embedded in text document. The search result is optimized by clustering and ranking documents according to the embedded dates. Allan et al. [24] performs news summarization by extracting ranked sentences from the news article. Swan and Allan [25] present a statistical model that extract important features from text and build overview timelines. We refer the interested reader to Alonso et al. [26], [27] for more details on the topic.

III. Design Strategy

Linea takes as input a set of unstructured texts describing historical events and which contains temporal information, namely dates. In this paper, we use Wikipedia articles as our source of unstructured text. The main reason for choosing Wikipedia is the sheer number of articles available: nearly 5 million articles in English.

Building a timeline is a simple two step process: first one collects the timeline events and then builds the timeline itself. Thus, a user must be able to accomplish two tasks: *A.* to explore important events about a topic that possibly spans a large timescale; and *B.* collect the desired events and build timelines. *Linea* was designed to solve these two issues.

A. Exploring Important Events on Large Time Scale

Signaling hidden content Discovering trends and features are common tasks in data-visualization [28]. Here we are interested in revealing hidden content. This is important to allow the user to see information that is potentially relevant but not obvious. We signal hidden events in the timeline itself by placing reminders between timeline events. More specifically, keywords are extracted from the hidden data and depicted between events of the timeline (see the hashtags placed in Figures 1 (left) and 6).

We also need to signal important time periods within the events. As an example, consider the top-left histogram cell shown in Figure 1. That histogram cell contains all the data starting from 3500 BC until 1991. Clearly, most of the events in the history of science happened in the last few centuries. Because the events are not distributed uniformly, we should provide tools that indicate potentially interesting content hidden in certain time intervals.

There are techniques available to explore time series data. Stack Zooming [29] and Continuum [30], for example, are designed to explore large scale time series data. They provide a simple way build a hierarchy in time, dynamic drill-down, and compare events in different intervals. While they can be used to navigate the time series, they do not provide indications of interesting aspects of the data that may be hidden. A solution based on dynamic drill-down requires the user to first select a small slice of data before noticing whether any interesting feature is present. We opt for showing as many important time intervals simultaneously as the user requires. Once the user select an interval, our event matrix automatically detect interesting cluster of events and it generates multiple histograms based on those clusters.

Explore large time scales The history of topics such as "Egypt" may span several millennia. Tools such as Continumm are capable of dealing with large scale time series. As in Continuum, our tool allow the user to specify the interval of

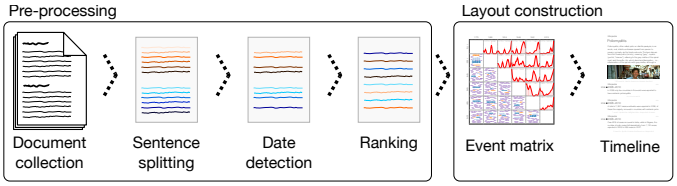


Fig. 2. *Linea* pipeline. Event detection is accomplished by sentence splitting and date detection. The event ranking is used to build default timelines. Event matrix and timeline construction. The event matrix is used to navigate through a story and build a timeline.

interest by dragging the mouse into the period of interest. This action will ignore data outside the selection to focus the user attention on the selected time period. Additionally, the user can specify the number n of interesting regions and let our system define n intervals automatically.

Smart defaults A timeline depicting all events associated with a subject can quickly become overwhelming, demanding great effort from users to filter out irrelevant and uninteresting information. Smart defaults can help with this task. Siegel and Heer identified smart defaults as an under-utilized technique that provide an initial representation of the data [31]. The authors argue that smart default can provide a stimulating starting point for users to explore the tool. We include smart default on *Linea* by ranking events and showing only the n most important ones.

B. Event Selection and Timeline Construction

Chronologically organized snippets A major requirement in our context is to provide a visualization from which users can easily read a chronological sequence of events. Unlike most of the representations available in the visualization literature of text summary, which uses complex geometry to summarize content, our summaries are visualized as vertical timelines. This is decision is motivated some evidence of the user preference for vertical scroll [32].

Interactive editing Lastly, *Linea* provides several interaction resources that allow users to hide visible snippets, make hidden snippets visible, and edit the textual summary in each snippet, thus customizing the timeline altogether.

In the following section, we detail the technicalities built into *Linea* and how they are implemented in our visualization system.

IV. The Linea Construction

Linea's pipeline can be divided into *data pre-processing* and *layout construction*. As illustrated in Figure 2, the pre-processing modulus comprises event detection and ranking. The layout construction comprises the event matrix, timeline, and interaction tools. In the next sections, we detail each of these moduli and substeps involved in the *Linea* construction. In this work, we do not devise a new approach to detect and evaluate events, but instead we rely on previous work, such as Alonso [33] and Alonso et al. [34]. As such, our technique benefits from advances in the information retrieval and natural language processing fields.

A. Data Pre-Processing

Our technique assumes as input a collection of unstructured text containing *implicit*, temporal information. By "implicit" we mean that explicit temporal information extracted from metadata (such as date of creation or last modification of a document) is not used; instead, we use *latent* temporal information, *i.e.*, temporal information written in the body of the text. This is an important distinction between our approach and many previous work. Unstructured texts that are not rich in temporal information will not benefit from this approach. Given a collection of unstructured text as input, the first task in the pre-processing step is to detect events, which are then ranked according to their importance. These two steps are described as:

Detecting events Following Luo et al. [10], the term "event" is used to characterize an event that happened in a particular time t. We differ, however, in how to detect it. Luo et al. detect events indirectly by tracking the number of documents and describing a happening in a certain time period, relying on metadata to achieve this goal. Our approach, on the other hand, uses dates written in the document body as proxy for events. The rationale is the following: because someone decided that it is worth writing that "something" happened at "sometime", the "something" is likely to be relevant. Therefore, in contrast to Luo et al. scheme, which is meant to track more general themes, our approach uses a finer grain definition of event.

Events are derived from sentences detected in the documents. To detect sentences, the input text is first subdivided into tokens. Then, sentence boundaries are detected by looking up special tokens (such as "!" or ".") that are not part of other tokens. Heuristics can be used to resolve issues such as punctuation inside quotes and other complications. We use the Stanford Tokenizer for sentence splitting [35].

The next step is to identify sentences that contain dates to define an event. A sentence may contain four types of temporal information [27]: date, time, duration, and sets. Date and time refer to particular instances of time ("4th of July 2014" and "9am", respectively), duration refers to time length ("ten years"), and sets refer to periodical events ("every 4th of July"). Our approach only considers "firm" dates, which are normalized to (year), (year-month), or (year-month-day) format. Duration is taken into account if it exceeds 10000 years ago. Throughout this paper, we use the word "date" to refer to these two cases only.

Ranking Events Ranks define the importance of an event in the timeline. To find the rank of an event, we first connect similar events, thus building a graph where events are nodes and edges are the connection between them. PageRank algorithm is then used to find "popular" events, thus increasing their rank.

Specifically, the ranking mechanism implemented in *Linea* is similar to the one employed in LexPageRank [22] and it consists of three steps: bag-of-words extraction, stochastic matrix construction, and the ranking computation itself. Let $\mathcal{A} = \{a_1, \ldots, a_m\}$ be a set of m events and W be the

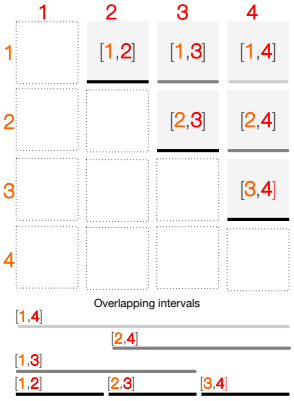
matrix where each row w_i^r corresponds to an event in $\mathcal{A}$ and each column w_j^c corresponds to a word in $\mathcal{A}$ (stop words are not considered; words are stemmed before building W). Entries w_{ij} in W are given by $\mathrm{tf}-\mathrm{idf}$ statistics [36], [37] defined as $w_{ij} = \mathrm{tf}(a_i, w_j) \times \mathrm{idf}(w_j, \mathcal{A})$, where $\mathrm{tf}(a_i, w_j)$ is the number of occurrences of the word w_j in a_i and $\mathrm{idf}(w_j, \mathcal{A}) = \log(\frac{m}{|\{a_k \in \mathcal{A} | w_j \in a_k\}|})$ ($|\cdot|$ is the cardinality of the set) accounts for how frequent w_j is across all events in $\mathcal{A}$.

From W we build the adjacency matrix M using a combination of k-nearest neighbors and cosine similarity measure. More precisely, given an event a_i we compute the $d_i \leq k$ ($k = 10$ in our implementation) most similar events a_j such that $< w_i^r, w_j^r > /(||w_i^r|| \, ||w_j^r||) > \alpha$, where $< \cdot, \cdot >$ accounts for the dot product and α is an input parameter (we use $\alpha = 5 \times 10^{-2}$ in our experiments). For each a_j selected as a neighbor of a_i we set $m_{ij} = 1/d_i$, making the remaining entries in the i-th row of M equal zero. If $d_i = 0$, all entries in the i-th row of M is set to $m_{ij} = 1/m$. Matrix M is a stochastic matrix, hence, the rank of each event a_i can be obtained by the well-known PageRank algorithm [38].

B. Layout Construction

Several visualization techniques can be used to visualize a set of ranked events [39], [10]. However, although previous works provide good visualization metaphors, they do not tackle the problem of handling an event set spanning large periods of time (centuries or even millennia). In this section, we present the *event matrix*, an approach for visualizing large time periods in multiple scales.

Event matrix The event matrix E is a set of histograms spanning distinct intervals of time (see inline image). Each row i contains the *start date* t_i whereas each column j contains the *end date* t_j of the histogram horizontal axis. Because $t_i \leq t_j$ for all $i \leq j$, the event matrix is strictly upper diagonal. Let E be a $n \times n$ event matrix and $E(i, j) = E(t_i, t_j)$ be the event histogram at cell (i, j). The histograms in a row $E(i, \cdot)$ have a fixed start date t_i, and, analogously, $E(\cdot, j)$ have fixed end date t_j. $E(1, n)$ includes all events extracted from a dataset. The set $\{E(i, i+1) \mid i < n\}$, *i.e.*, the histograms just above the matrix diagonal, covers disjoint time intervals. For all cells (i, j), $(t_i, t_j) \subset (t_i, t_{j+1})$, *i.e.*, the time interval of a given matrix cell is contained in its right adjacent matrix cell's interval. The inline image illustrates the event matrix for the time interval $[1, 4]$. The bottom lines show the intervals represented in the histogram matrix: the matrix diagonal contains the narrowest, non-overlapping, intervals; the intervals overlap and become broader as cells get closer to $E(1, 4)$. In this paper, we use the same number of bins for all histograms in E.



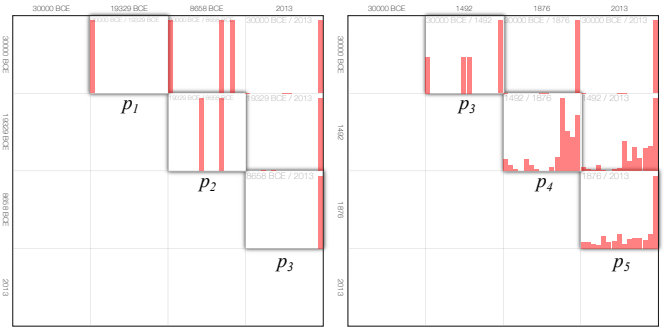


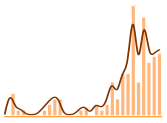
Fig. 3. The 4×4 event matrices of US history show two approaches for the subdivision of the interval $[30000\ \mathrm{BCE}, 2013]$. Left: uniform subdivision. The highlighted histograms cover intervals of approximately 10000 years: $p_1 = [30000\ \mathrm{BCE}, 19329\ \mathrm{BCE}]$, $p_2 = [19329\ \mathrm{BCE}, 8658\ \mathrm{BCE}]$, and $p_3 = [8658\ \mathrm{BCE}, 2013]$. Right: non-uniform subdivision. p_3, p_4, and p_5 cover approximately 31000, 380, and 137 years. The non-uniform subdivision reveals more interesting details using the same matrix size. See Section IV-B for details on how to find appropriate interval breaks.

Break selection We build the event matrix E by determining the start and ending break dates $I = \{t_1, \ldots, t_n\}$ for each of its rows and columns. One option is to let users control the interval breaks they are interested in by manually selecting dates. Nevertheless, it is important to have automated default ranges so as to reveal interesting patterns to users.

A straightforward approach is to subdivide the interval uniformly. The problem with uniform subdivision is that the larger the time span the more difficult it is to visualize details. For instance, when visualizing the event matrix of the history of the US, it becomes harder to get a meaningful overview of the data (see left matrix in Figure 3). Since our goal is to show both an overview and interesting details, we use a different approach for building interval ranges.

We cluster all events according to their time-stamp using $k-$means. Since $k-$means will cluster data in 1D, it can be done optimally. Then we set the dates t_i as the lower time-stamp of each each cluster. This simple strategy naturally concentrates finer intervals in denser time intervals, thus allowing a better visualization of details and event distribution. Figure 3 compares event matrices built with uniformly distributed and cluster-based intervals. Notice how details show up when using the clustering scheme.

Smooth curves Instead of showing the histograms directly, we use a B-splines to approximate its shape (see inline figure). A smooth curve draws more attention to peaks and valleys of the histogram than the histogram themselves.



Alternative layouts to the event matrix An alternative approach for generating histograms of progressively smaller, nested, intervals is binary subdivision. The main disadvantage of binary subdivision is the exponential growth in the total number of intervals generated. However, the problem is mitigated if the number of subdivision is low. H-trees [40] make better use of the space by means of a fractal-like subdivision. However, H-trees are mainly designed to present

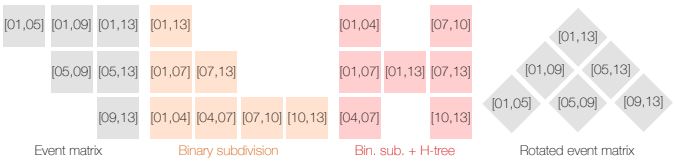


Fig. 4. Layouts displaying the uniform subdivision of the time interval $p = [2001, 2013]$. The event matrix has compact layout; binary subdivision generates more intervals; and H-tree has compact layout using the same intervals as binary subdivisions. As H-tree grows, "children" intervals are positioned further away from "parent" interval.

hierarchies (e.g., parents/children relation), thus it does not provide an intuitive way of representing the relation between sub-intervals.

C. Building Timelines Interactively

We now show how previous concepts are tied together in our approach.

Timeline The timeline is where events chosen by the user (or highly recommended by the ranking algorithm) are depicted in a vertical arrangement. The timeline is divided into three parts: a summary of the subject of interest, hidden event snippets, and visible event snippets (see Figure 5). The summary on the top of the timeline is an (optional) abstract that can be assigned to a story. Since our main source of data is Wikipedia, we use the abstract data provided with each article as the story summary. An image can be manually attached to an event.

Hidden snippets are flagged by a list of keywords extracted from the hidden content. This list is shown between visible snippets respecting the time stamp of the hidden events. An example of hidden content is shown in Figure 5 between the dates 1879 and 1905, where one can read `18 events: #einstein #marić #paper #published`. We use a simple word count to retrieve the most frequent words in the hidden events and display them as hashtags. By clicking on hidden snippets icon, the hidden event with the highest priority is shown, becoming part of the timeline. Visible event snippets contain the name of the data source, and optional title, the description of the event, and an optional image.

Default timeline A subject may contain dozen, hundreds, or even thousands of events. Presenting all events in a timeline is overwhelming, as the user can face difficulties finding the focus of interest amidst many available events. Random sampling is not a viable alternative as the selected snippets may not be representative of the whole. We use the ranking introduced in Section IV-A to automatically build default timelines depicting the top ranked events. Users can then add/remove events deemed important/irrelevant. Event markers (the blue dots in Figures 1 and 6) are shown in the event matrix to highlight from which time interval the user is picking out new events to compose the timeline. This is useful to help users decide whether all the important time intervals are represented with enough events.

Filtering *Linea* uses brushing and linking to allow users to explore the event matrix. By selecting a time interval s in

Fig. 5. Timeline components. The story summary (top), the hidden events are shown between visible event snippets. Snippets are highlighted when the mouse cursor goes over it allowing the user to expand an image, if any, and hide the events.

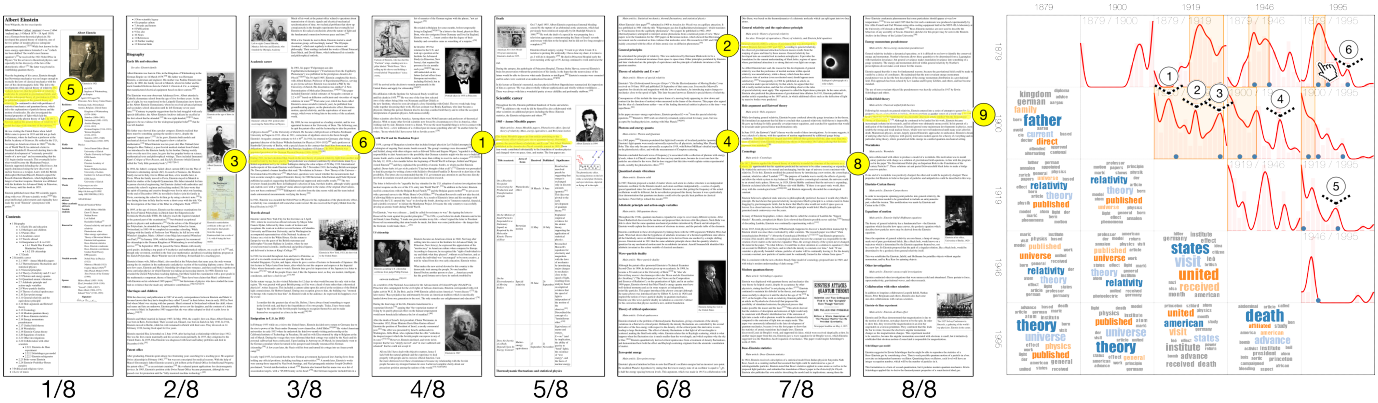


Fig. 6. The Wikipedia article entitled "Albert Einstein" (shown here across eight panels) contains over 300 sentences. The rightmost image shows a 5×5 event matrix of that article. The yellow circles show the nine events with highest priority (see Section IV-A). Numbers represent the order in which events appear in the timeline (from old to new). Yellow pen markers show the position in the article from which these events were extracted. The following keywords were omitted from the word cloud: *albert* and *einstein*.

one matrix cell, s is reflected to all other cells that contain that interval. This allow users to inspect a time period in different level of details, which may reveal interesting patterns. In addition, when a time interval is selected, all events that make up the tailored timeline are temporarily replaced by events within the selected interval. The user can then browse all events for that particular interval, adding those of interest to the timeline. By deselecting the cell, the tailored timeline is restored in addition to any new events chosen by the user.

D. Implementation

Linea is implemented using JavaScript, C++, and Java. The front end relies on D3 [41] and plugins to render the event matrix and timeline. The event extraction algorithm relies on Stanford CoreNLP [35]. In this paper, our source of unstructured temporal text is the English version of Wikipedia (2013/05/03 version). Each Wikipedia article was pre-processed to remove wiki tags. The removal of wiki tag is done on the backend along side with the event extraction and ranking. *Linea* also relies on DBpedia for extracting (optional) article lead [42].

V. Case Study

In this section, we present one case study for *Linea*, namely, Albert Einstein's life. More case studies can be found in the supplemental material. In this case study we employ *Linea* to visualize Albert Einstein's life. The input data come from Wikipedia's *Albert Einstein* article. Here we use a single article, but in practice, others could be included, *e.g.*, *Einstein family*, *History of special relativity*, etc. The article contains over 330 sentences, of which our pre-processing step selected 85 events. Although the number of events collected is nearly a fourth of the input data, a timeline with 85 events is too dense.

Figure 6 shows how *Linea* assists the summarization of documents. When an article is loaded for the first time, the default timeline is built based on the ten highest ranking events. Nine events ranked as the most relevant correspond to Einstein's scientific career: special theory of relativity and Brownian motion (1905); general theory of relativity (1907, 1916); and Einstein's prediction that light would be bent by the Sun's gravity (1911). This example shows that the default timeline is a good starting point for a deeper investigation. Notice from Figure 6 that the events are not organized linearly in the article, but they go back and forth (shown as numbered yellow circles). This is expected because the article is organized by subject instead of chronologically; hence, each section may cover large periods of time. *Linea* chronologically reorganizes the information available on Wikipedia. Figure 5 shows Einstein's timeline after we edited it (in total 6 events were removed — two of them were redundant — and 2 added). Note how the chosen events condense the whole article.

Out of the ten default events, only one was not related to Einstein's scientific career, namely, his death (last event shown in Figure 5). Although the date of his death was extracted from Wikipedia, its description was built automatically. This is a limitation of the text analysis performed. The Wikipedia article reads:

> On 17 April 1955, Albert Einstein experienced internal bleeding caused by the rupture of an abdominal aortic aneurysm, which had previously been reinforced surgically by Dr. Rudolph Nissen in 1948. [...] He died in Princeton Hospital early the next morning at the age of 76, having continued to work until near the end.

This example illustrates the challenge of retrieving events from raw text; an important event, his death, was not extracted because our text analysis cannot process relative dates: the "next morning" is in relation to 17 April 1955. This problem was mitigated by detecting that the subject of interest is a person and extracting information typically found in a person's life, such as birth, death, marriage, and others, from Wikipedia's infobox. In this case, his death date was extracted and an event was created based on it.

The 5×5 event matrix in Figure 6 shows the distribution of events over time. The highest peak in the top right histogram

(1879/1995) ranges from 1900 to 1919 (see selection in the top-right matrix cell), which suggests an important period in Einstein's life. By selecting that interval, *Linea* displays only event snippets in that range, which in this case, is mostly related to Einstein's scientific career. Moreover, since the selection is reflected in other time scales, the event matrix reveals three smaller peaks, labeled as (1), (2), and (3) in the matrix cell (1900/1919). With further exploration, the user can observe that the three peaks correspond to (1) family matters (marriage and children) and special theory of relativity, (2) academic career, and (3) family matters (divorce and new marriage) and the general theory of relativity.

The selection around the highest peak is not the only range containing interesting features. A small inflection in the curve outside the selection of the (1879/1995) cell, turns out to be another peak inside the (1900/1946) cell, labeled as (4). By selecting that area, the timeline reveals that most of the events correspond to Einstein's emigration to the US. The word cloud summarizes it nicely as the words "United States" and "Visit" are prominent. Note that the histogram essentially flattens out after 1960, just after Einstein's death (1946/1995), except by small peaks. Lastly, by exploring the period around his death (labeled as (6)), a curious event is revealed:

> After the death of Israel's first president, Chaim Weizmann, in November 1952, Prime Minister David Ben-Gurion offered Einstein the position of President of Israel, a mostly ceremonial post.

Once a user has explored the collection of events, she/he can build a tailored timeline.

VI. User feedback

We performed a qualitative user evaluation [43] to get feedback about the usefulness and usability of *Linea*. A total of 12 users (2 female 10 male) volunteered to participate in a 2-hour session from a graduate center. Volunteers were able to read English and were enrolled in computer science undergraduate or graduate programs. Users had no previous experience with *Linea*.

First, we demoed *Linea*'s features. All users were encouraged to ask questions to assure a good understanding of how to navigate *Linea*. Second, users were asked to use the tool to explore different pre-specified datasets: a training dataset (92 events), World War II (666 events), and poliomyelitis (194 events), and create a timeline that they deemed relevant, keeping track of the amount of time needed to complete the tasks. Users were asked an open-ended question on their overall impressions and opinions (positive, neutral or negative). At the end, we asked whether they agreed with two statements "I had previous knowledge about the topic" and "I learned something about the topic". Answers were collected via a 5-point Likert scale (strongly agree — strongly disagree). The users spent 23 minutes on average during the training phase. The average time spent to build the timelines, World War II and poliomyelitis, was 17 and 10 minutes, respectively. The average number of events in the final timeline were 17.17 (training), 44.91 (WWII), and 26.33 (polio). Half of the users reported having previous knowledge ("agree" or "strongly agree") on the topic of WWII, while no one reported previous knowledge on the topic of poliomyelitis ("disagree" or "strongly disagree").

Written, free-text, overall impressions and user feedback were collected. Both positive and negative criticisms were provided. Besides providing their opinion about *Linea*'s usefulness and usability, the volunteers were encouraged to point out improvements as to general usability. The given suggestions included to highlight a word in the word cloud when it is selected and to allow users to click anywhere to remove selection from the event matrix. Volunteers also suggested new features to be incorporated, for instance, to enable queries using keywords and add weights to words based on previous knowledge. Overall impressions were positive. The features users found most useful were the easy to use event matrix, the ability to remove words from the event matrix to filter out the word cloud, and the brushing mechanism, which users found to be efficient to select events from specific time intervals. Users also reported that changing matrix resolution dynamically was useful for exploring all events. Most users reported that they were able to uncover interesting facts for all datasets they investigated. For all datasets, most users reported have gained knowledge on the topic ("agree" or "strongly agree"), by exploring the timelines.

VII. Discussion and Conclusion

Our case study showed that relevant events appear as peaks in the event matrix cells (see more case studies in the supplemental materials). Peaks in coarser time scales can represent a set of important events that show up as multiple peaks in finer scales, evidencing the importance of providing visual representations in different level-of-details. The user feedback about the selection tool implemented in *Linea* was positive. It allows users to easily explore events in the different time scales while showing clues of important time periods. Word clouds are also helpful for providing summaries of time intervals. Additionally, we are investigating alternatives to incorporate information extracted from the histograms into the ranking mechanism to provide even better default timelines. In the current state, the most time-consuming step is retrieving the context around an event, which was also pointed out by users in the informal evaluation session, thus being an aspect for further investigation.

Although dates have proven to be a good proxy for interesting events, this is not always the case. Sentences describing important events may not contain dates suitable to *Linea* (*e.g.*, "4th of July is an important american holiday") or may not contain time-tags at all. An interesting polio-related event in Wikipedia reads:

> Ancient Egyptian paintings and carvings depict otherwise healthy people with withered limbs, and children walking with canes at a young age.

In this case, time is implicitly defined as "Ancient Egypt". Conversely, some irrelevant events may be tagged. Although false-positives and false-negatives may occur, in general, the

assumption that dates are good proxy for events has performed well on our experiments.

The event matrix has a similar disadvantage as scatterplot matrices and others: as the matrix dimensions increase, it becomes harder to navigate through the cells and explore the data. This is mitigated by the fact that the user can select a particular time period, effectively filtering out all the data outside the specified range, thus reducing the number of dimensions required to navigate through the data. The disadvantage, however, of this workaround is that the user will no longer have overview of the whole data. This is a drawback we should approach in a future work.

Making use of a new visualization widget called event matrix, *Linea* turned out to be quite effective in enabling tools to explore collections of time-stamped events in different time scales. The provided case studies and user feedback showed *Linea* is an easy to use method that can benefit professionals from different fields.

ACKNOWLEDGMENT

This research was partially supported by FAPESP–Brazil (grant #2011/22749-8).

REFERENCES

[1] J. Alleman and J. Brophy, "History is alive: Teaching young children about changes over time," *The Social Studies*, vol. 94, no. 3, pp. 107–110, 2003.

[2] F. Perraudin, A. Mason, and D. Louter, "Digital storytelling: an interactive timelines project," http://www.theguardian.com/info/digital-journalism-scheme-blog/2014/oct/22/digital-storytelling-an-interactive-timelines-project#img-2, accessed April 13, 2015.

[3] W. Mullër and H. Schumann, "Visualization methods for time-dependent data-an overview," in *IEEE Winter Sim Conf*, vol. 1, 2003, pp. 737–745.

[4] S. Laxman and P. S. Sastry, "A survey of temporal data mining," *Sadhana*, vol. 31, no. 2, pp. 173–198, 2006.

[5] W. Playfair, *Playfair's commercial and political atlas and statistical breviary*. Cambridge University Press, 2005.

[6] S. Havre, B. Hetzler, and L. Nowell, "Themeriver: Visualizing theme changes over time," in *IEEE INFOVIS*, 2000, pp. 115–123.

[7] S. Havre, E. Hetzler, P. Whitney, and L. Nowell, "Themeriver: Visualizing thematic changes in large document collections," *IEEE TVCG*, vol. 8, no. 1, pp. 9–20, 2002.

[8] W. Cui, S. Liu, L. Tan, C. Shi, Y. Song, Z. Gao, H. Qu, and X. Tong, "Textflow: Towards better understanding of evolving topics in text," *IEEE TVCG*, vol. 17, no. 12, pp. 2412–2421, 2011.

[9] S. Liu, M. X. Zhou, S. Pan, Y. Song, W. Qian, W. Cai, and X. Lian, "Tiara: Interactive, topic-based visual text summarization and analysis," *ACM TIST*, vol. 3, no. 2, pp. 25:1–25:28, 2012.

[10] D. Luo, J. Yang, M. Krstajic, W. Ribarsky, and D. A. Keim, "Eventriver: Visually exploring text collections with temporal references," *IEEE TVCG*, vol. 18, no. 1, pp. 93–105, 2012.

[11] Y. Tanahashi and K.-L. Ma, "Design considerations for optimizing storyline visualizations," *IEEE TVCG*, vol. 18, no. 12, pp. 2679–2688, 2012.

[12] S. Liu, Y. Wu, E. Wei, M. Liu, and Y. Liu, "Storyflow: Tracking the evolution of stories," *IEEE TVCG*, vol. 19, no. 12, pp. 2436–2445, 2013.

[13] S. Rose, S. Butner, W. Cowley, M. Gregory, and J. Walker, "Describing story evolution from dynamic information streams," in *IEEE VAST*, 2009, pp. 99–106.

[14] P. Xu, Y. Wu, E. Wei, T.-Q. Peng, S. Liu, J. J. Zhu, and H. Qu, "Visual analysis of topic competition on social media," *IEEE TVCG*, vol. 19, no. 12, pp. 2012–2021, 2013.

[15] K. Wongsuphasawat, J. A. Guerra Gómez, C. Plaisant, T. D. Wang, M. Taieb-Maimon, and B. Shneiderman, "Lifeflow: visualizing an overview of event sequences," in *Proc of the ACM CHI*, 2011, pp. 1747–1756.

[16] C. Plaisant, R. Mushlin, A. Snyder, J. Li, D. Heller, and B. Shneiderman, "Lifelines: using visualization to enhance navigation and analysis of patient records," in *Proc of the AMIA*, 1998, p. 76.

[17] K. Wongsuphasawat and D. Gotz, "Exploring flow, factors, and outcomes of temporal event sequences with the outflow visualization," *IEEE TVCG*, vol. 18, no. 12, pp. 2659–2668, 2012.

[18] J. A. Fails, A. Karlson, L. Shahamat, and B. Shneiderman, "A visual interface for multivariate temporal data: Finding patterns of events across multiple histories," in *IEEE VAST*, 2006, pp. 167–174.

[19] A. A. Bui, D. R. Aberle, and H. Kangarloo, "Timeline: Visualizing integrated patient records," *IEEE TITB*, vol. 11, no. 4, pp. 462–473, 2007.

[20] C. Stab, K. Nazemi, and D. W. Fellner, "Sematime-timeline visualization of time-dependent relations and semantics," in *Adv in Vis Comput*. Springer, 2010, pp. 514–523.

[21] P. André, M. L. Wilson, A. Russell, D. A. Smith, A. Owens *et al.*, "Continuum: designing timelines for hierarchies, relationships and scale," in *ACM UIST*, 2007, pp. 101–110.

[22] G. Erkan and D. R. Radev, "Lexpagerank: Prestige in multi-document text summarization." in *EMNLP*, vol. 4, 2004, pp. 365–371.

[23] J. Strötgen, O. Alonso, and M. Gertz, "Identification of top relevant temporal expressions in documents," in *Proc of the TempWeb 2012*. ACM, 2012, pp. 33–40.

[24] J. Allan, R. Gupta, and V. Khandelwal, "Temporal summaries of new topics," in *Proc of the 24th ACM SIGIR*. ACM, 2001, pp. 10–18.

[25] R. Swan and J. Allan, "Automatic generation of overview timelines," in *Proc of the 23th ACM SIGIR*. ACM, 2000, pp. 49–56.

[26] O. Alonso, M. Gertz, and R. Baeza-Yates, "On the value of temporal information in information retrieval," *SIGIR Forum*, vol. 41, no. 2, pp. 35–41, 2007.

[27] O. Alonso, J. Strötgen, R. Baeza-Yates, and M. Gertz, "Temporal information retrieval: Challenges and opportunities," in *Proc of the TWAW 2011*, 2011, pp. 1–8.

[28] T. Munzner, *Visualization Analysis and Design*. CRC Press, 2014.

[29] W. Javed and N. Elmqvist, "Stack zooming for multi-focus interaction in time-series data visualization," in *IEEE PacificVis*, 2010, pp. 33–40.

[30] P. André, M. L. Wilson, A. Russell, D. A. Smith, A. Owens, and m. schraefel, "Continuum: Designing timelines for hierarchies, relationships and scale," in *Proc of the 20th ACM UIST*, 2007, pp. 101–110.

[31] E. Segel and J. Heer, "Narrative visualization: Telling stories with data," *IEEE TVCG*, vol. 16, no. 6, pp. 1139–1148, 2010.

[32] J. Nielsen, "Scrolling and scrollbars," http://www.nngroup.com/articles/scrolling-and-scrollbars/, 2005, accessed April 13, 2015.

[33] O. R. Alonso, "Temporal information retrieval," Ph.D. dissertation, University of California, Davis, 2008.

[34] O. Alonso, R. Baeza-Yates, and M. Gertz, "Exploratory search using timelines," in *SIGCHI 2007 Workshop on Exploratory Search and HCI Workshop*, no. 1, 2007, pp. 1–4.

[35] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proc of the HLT-NAACL 2003*. Association for Computational Linguistics, 2003, pp. 173–180.

[36] J. Ramos, "Using TF-IDF to Determine Word Relevance in Document Queries," Dept of CS, Rutgers University, Tech. Rep., 2003.

[37] A. Rajaraman and J. D. U., *Mining of Massive Datasets*. Cambridge University Press, 2011.

[38] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep. 1999-66, 1999.

[39] M. Krstajic, E. Bertini, and D. Keim, "Cloudlines: Compact display of event episodes in multiple time-series," *IEEE TVCG*, vol. 17, no. 12, pp. 2432–2439, 2011.

[40] I. Herman, G. Melancon, and M. Marshall, "Graph visualization and navigation in information visualization: A survey," *IEEE TVCG*, vol. 6, no. 1, pp. 24–43, 2000.

[41] M. Bostock, V. Ogievetsky, and J. Heer, "D3 data-driven documents," *IEEE TVCG*, vol. 17, no. 12, pp. 2301–2309, 2011.

[42] P. N. Mendes, M. Jakob, and C. Bizer, "Dbpedia for nlp: A multilingual cross-domain knowledge base," in *Proc of the LREC*. ELRA, 2012.

[43] H. Lam, E. Bertini, P. Isenberg, C. Plaisant, and S. Carpendale, "Empirical studies in information visualization: Seven scenarios," *IEEE TVCG*, vol. 18, no. 9, pp. 1520–1536, 2012.