TiCCo: Time-Centric Content Exploration

Philip Hausner*
hausner@informatik.uni-heidelberg.de
Institute of Computer Science,
Heidelberg University
Heidelberg, Germany

Dennis Aumiller*
aumiller@informatik.uni-heidelberg.de
Institute of Computer Science,
Heidelberg University
Heidelberg, Germany

Michael Gertz
gertz@informatik.uni-heidelberg.de
Institute of Computer Science,
Heidelberg University
Heidelberg, Germany

ABSTRACT

Time is a natural way to order information and can be utilized to summarize events and to construct a chronology of contents within a document collection in many application domains. Structuring the sequence of events along a timeline allows users to grasp information at-a-glance, which enables them to get familiar with a topic in only a short amount of time and can hence support the analysis of more complicated and heterogeneous textual data. The manual construction of timelines, however, is a tedious and error-prone task, leading to static timeline representations that limit users to a passive role. In this paper, TiCCo, an automated extraction pipeline from arbitrary English and German text collections, is provided and presented to the user in an interactive manner. This puts the user in an active role in which she not only absorbs knowledge, but also influences in which ways the information is presented to her. In-depth investigations of a specific point in time are augmented by utilizing time-centric co-occurrence graphs that further summarize information extracted from a document collection, and enable users to explore the chronology of events by allowing them to interact with the constructed graphs as well as the underlying documents.

CCS CONCEPTS

• **Information systems → Document representation**.

KEYWORDS

text analytics, temporal information, text exploration, automatic timeline generation, co-occurrence graph

ACM Reference Format:
Philip Hausner, Dennis Aumiller, and Michael Gertz. 2020. TiCCo: Time-Centric Content Exploration. In *Proceedings of CIKM '20: Conference on Information and Knowledge Management (CIKM '20 (submitted for review))*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

As the amount of documents and textual data in general is steadily increasing, average users as well as professionals are faced with an abundance of available data. However, it is usually unfeasible for users to read an entire document collection to grasp all necessary information, and even for professionals it is often beneficial to get an initial overview of a new topic before contemplating the details. Moreover, concise data representation can accelerate the work process in general, e.g., by enabling users to quickly confirm or reject certain assumptions they have about the data. With that in mind, representing textual data regarding temporal information contained in the text is often a useful approach to organize information. This is true for various domains: In the legal domain, it can help to build case chronologies; in the news domain, it can summarize important political events; and in the historical domain, it can give an overview of the events during a certain period of time. A typical way to visualize such chronological data is to employ timeline representations that give a concise overview of a time period as well as it is a well-known and intuitive tool many users are already familiar with. In this paper, we demonstrate TiCCo, a novel system to organize and explore document collections in a time-centric fashion utilizing a previously introduced graph-based approach. Thereby, we provide the following contributions:

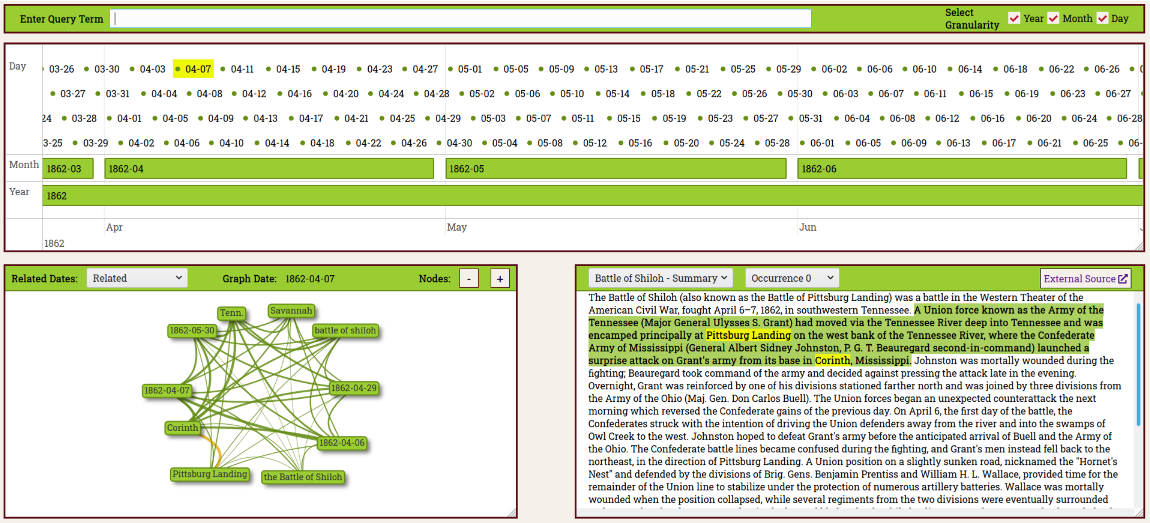
- An extendable pipeline to automatically annotate and extract time-centric co-occurrence graphs from English or German document collections.
- An interactive user frontend that allows for an intuitive exploration of large document collections in a time-centric manner. The system includes references to the original text excerpts as well as query capabilities. Figure 1 gives a first impression of the interface.
- An open-source implementation of the proposed pipeline available on GitHub[1], as well as a publicly accessible demo[2] and video demonstration[3].

Another important key aspect of this work is the exploratory component of the proposed system. The goal is to not only provide the user with information, but also to encourage her to actively interact with the components of a timeline. Therefore, constructed graphs contain links to the underlying documents from which the graph was built. This enables users to understand the origin of nodes and edges in the graph and gives them opportunities for further explorations as well as a deeper understanding of the applied method. Further exploratory possibilities include the querying for terms , or the access to related dates given a time-centric graph.
After a summary of related work in Section 2, Section 3 details the underlying model, and Section 4 introduces the proposed implementation based on documents from the English Wikipedia related to the American Civil War.

*Both authors contributed equally to this research.

CIKM '20 (submitted for review), October 19–23, 2020, Galway, Ireland

[1] https://github.com/PhilipEHausner/TiCCo
[2] https://ticco.ifi.uni-heidelberg.de
[3] https://www.youtube.com/watch?v=z_tNHKxanak



The Battle of Shiloh (also known as the Battle of Pittsburg Landing) was a battle in the Western Theater of the American Civil War, fought April 6–7, 1862, in southwestern Tennessee. A Union force known as the Army of the Tennessee (Major General Ulysses S. Grant) had moved via the Tennessee River deep into Tennessee and was encamped principally at Pittsburg Landing on the west bank of the Tennessee River, where the Confederate Army of Mississippi (General Albert Sidney Johnston, P. G. T. Beauregard second-in-command) launched a surprise attack on Grant's army from its base in Corinth, Mississippi. Johnston was mortally wounded during the fighting; Beauregard took command of the army and decided against pressing the attack late in the evening. Overnight, Grant was reinforced by one of his divisions stationed farther north and was joined by three divisions from the Army of the Ohio (Maj. Gen. Don Carlos Buell). The Union forces began an unexpected counterattack the next morning which reversed the Confederate gains of the previous day. On April 6, the first day of the battle, the Confederates struck with the intention of driving the Union defenders away from the river and into the swamps of Owl Creek to the west. Johnston hoped to defeat Grant's army before the anticipated arrival of Buell and the Army of the Ohio. The Confederate battle lines became confused during the fighting, and Grant's men instead fell back to the northeast, in the direction of Pittsburg Landing. A Union position on a slightly sunken road, nicknamed the "Hornet's Nest" and defended by the divisions of Brig. Gens. Benjamin Prentiss and William H. L. Wallace, provided time for the remainder of the Union line to stabilize under the protection of numerous artillery batteries. Wallace was mortally wounded when the position collapsed, while several regiments from the two divisions were eventually surrounded

Figure 1: Overview of the user frontend showing the main interface components. The underlying data set is extracted from Wikipedia articles covering the American Civil War. On top, the user can query for terms and adjust visibility of timeline granularities. The main timeline is depicted in the center. On the bottom left, the graph associated with 7th April 1862 is shown. In the bottom right corner, co-occurrences of nodes of interested can be investigated further.

2 RELATED WORK

Because time is one of the key dimensions of human life, the processing of temporal information contained in textual data has become a widely researched field of study. Recent surveys by Campos et al. [2] and Lim et al. [5] give an introduction to the field and outline its relevance for many challenges including topics like question answering and temporal text similarity. Moreover, timelines are generally acknowledged to be a suitable representation for temporal data, and a timeline summarization aims to give a compact overview of a topic by extracting sentences or short paragraphs from document collections that best describe points in time mentioned in the data. Recently, Steen and Markert [10] introduced an abstractive timeline summarization model that computes timelines completely unsupervised using multi-sentence-compression. However, in general this is not a suitable representation for exploratory scenarios, since there is no way for the user to interact with the result. On the other hand, research by Alonso et al. [1] has shown that timelines can be used for the exploration of search results, and recently, Piskorski et al. [6] have introduced a framework for timeline generation and visualization that allows users to explore content from online news. Spitz et al. [8] presented EVELIN, a graphical interface that shows in which ways word networks can be effectively utilized for exploratory tasks, however, their implementation does not leverage temporal information to structure the underlying data. Furthermore, Prytkova et al. [7] and Spitz et al. [9] introduced time-centric graph models similar to ours, but they did not formalize their approaches in the form of a timeline or provided ways to utilize them for exploratory scenarios.

The graph model employed in this work is based on Hausner et al. [3]. Based on their theoretical framework, we developed a pipeline implementation that proves how the proposed model can be used for effective content exploration.

3 THEORETICAL BACKGROUND

In this section, we briefly recap the theoretical framework by Hausner et al. [3] this work is based on. First, the underlying graph model is introduced, and second, a node ranking function is presented.

3.1 Document Model

Let $\mathcal{P}$ be a collection of documents (or *pages*). Each document $p \in \mathcal{P}$ consists of a set of sentences $s \in p$. The set of all sentences is denoted by $\mathcal{S} = \cup_{p \in \mathcal{P}} \{s | s \in p\}$, and each sentence $s \in \mathcal{S}$ is treated as a bag-of-words, such that the order of words is not preserved in this model. It should be noted that two sentences that contain identical words are both included in $\mathcal{S}$, and are not treated as the same sentence in this model. For temporal exploration of data, it is furthermore of utmost importance to identify words that carry temporal information. In this context, $\mathcal{D}$ denotes a set of dates with two dates $d_1 \in \mathcal{D}$ and $d_2 \in \mathcal{D}$ being equal if they carry the same temporal information, e.g., if they refer to the same day. To account for varying temporal granularities, $\mathcal{D}$ is partitioned into $\mathcal{D} = \mathcal{D}_y \cup \mathcal{D}_m \cup \mathcal{D}_d$, where the indices denote years, months, and days, respectively. Utilizing this partitioning, an inclusion hierarchy can be formulated: For each day, there exists a month in which it is included, and the same relation holds between months and years.

3.2 Time-Centric Co-Occurrence Graphs

Given a set of dates $\mathcal{D}$, a **time-centric co-occurrence graph** is a weighted graph $G_d = (N_d, L_d)$ with nodes N_d being the terms extracted in a window of x sentences around timestamp $d \in \mathcal{D}$, and links L_d representing co-occurrences between the terms in the same context window. The set of all time-centric co-occurrence graphs is denoted by $\mathcal{G}_{\mathcal{D}} = \{\mathcal{G}_d \mid d \in \mathcal{D}\}$. It is crucial to note that not for each occurrence of a date $d \in \mathcal{D}$, a separate time-centric graph exists, but co-occurrences around different instances of d are aggregated in the same graph G_d. Utilizing the introduced inclusion hierarchy, it can be stated that for each graph G_{d_1} representing the network for a day d_1, any node n or edge e that is part of G_{d_1}, also needs to be included in graph G_{m_1} for the month m_1 containing d_1. Again, the same holds for months and years, respectively.

3.3 Sentence Functionality

To extend the exploratory possibilities of the model, the sentence functionality is defined by **sent** : $L_d \to \mathbb{P}(\mathcal{S} \times \mathcal{S})$ with $\mathbb{P}$ being the power set. The sentence functionality assigns to each edge all pairs of sentences in which the two words that are connected by the edge co-occur. It should be noted that the sentences need not be distinct. This allows to not only indicate the relation between two words in a graph, but also links the edge to its origins, i.e., the mutual co-occurrences of two terms in the given document collection.

3.4 Node Weighting

Each node is assigned a weight computed by the *term frequency - inverse timestamp frequency (tf-itf)* weighting scheme, an adaption of the popular tf-idf scheme. It is defined by:

$$\text{tf-itf}(n, d, D) := \text{tf}(n, d) \cdot \text{itf}(n, D), \qquad (1)$$

with tf being the normalized term frequency in the context window around $d \in D$, and itf defined as

$$\text{itf}(n, D) = log\left(\frac{M}{1 + |\{d \in D : \text{tf}(n, d) > 0\}|}\right), \qquad (2)$$

with M being the number of unique timestamps in the document collection. The denominator is increased by 1 to avoid zero-division.

4 DEMONSTRATION

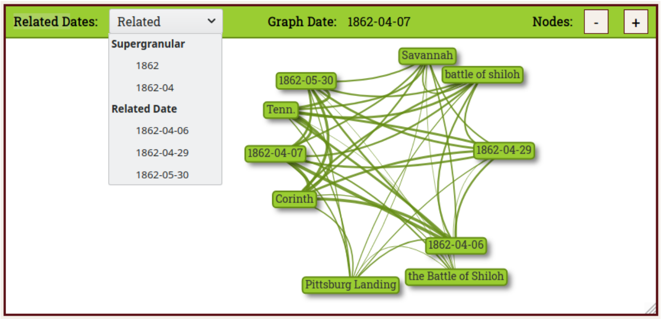
In this section, we demonstrate the utility of our proposed interactive user interface, especially with an application for downstream exploratory tasks. We further give an introduction to the preprocessing of data.

4.1 Preprocessing

In the preprocessing step, time-centric co-occurrence graphs are created from raw text. In a first step, temporal expressions are extracted from the document collection using the temporal tagger HeidelTime [11]. Since all preprocessing is programmed using Python, this work also provides potential users with a Python wrapper for HeidelTime along with an installation script [4]. The text is then further processed with the help of spaCy[4], which is used for sentence splitting, named entity recognition, lemmatization, and removal of out-of-vocabulary tokens, resulting in a set of sentences for each document. Time-centric co-occurrences are then extracted from this document structure, which are finally employed for the construction of the time-centric co-occurrence graphs.

4.2 Interactive User Interface

As shown in Figure 1, the user interface is divided into four major parts. A simple search interface and selector for visibility of the three different granularities, i.e., years, months and days. Further, a central timeline, accompanied by a graph interface, and lastly, a document browser that displays the source texts from which the graphs were generated. The timeline as well as all graphs are visualized with the help of the JavaScript library vis.js[5]. The individual items are not independent from each other, but are connected in various ways. For example, clicking on a date in the timeline opens the associated time-centric graph, and clicking on a node or edge in the graph displays the co-occurrences from which the node or edge was extracted in the document browser. An underlying MongoDB database provides graphs on demand, hence, making this approach applicable to data sets consisting of a large amount of graphs.



Related Dates: Related

Graph Date: 1862-04-07

Nodes: - +

Figure 2: The graph visualization interface including a drop-down menu for selection of related dates.

4.2.1 *Timeline.*

The timeline interfaces shows all dates for which a time-centric graph exists, grouping dates of different granularities. Dates of month and year granularity are displayed as a box ranging along the respective time frame, indicating that they do not refer to a single point in time, but to a longer period of time. To keep this interface clearly arranged at all times, day and month granularities are hidden if the user zooms out too far, e.g., days are only displayed if the timeline depicts a range of less than a year.

4.2.2 *Graph Visualization.*

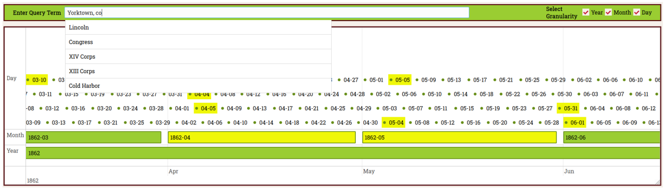
The graph interface displays the time-centric co-occurrence graph for a given date. Nodes represent terms occurring around a date, and edges co-occurrences between the two terms. Hence, the graph describes entities and events of a point in time as well as their relations. The interface also enables a user to select graphs from related dates, i.e., either dates of coarser granularity, or dates that are represented by a node in the graph. Figure 2 shows an example. It is also possible to increase and decrease the number of nodes displayed to broaden or narrow the scope of investigation. The displayed nodes are chosen based on their tf-itf scores. As previously mentioned, clicking on an element of the graph reveals co-occurrences of the element in the data set which are displayed in the document browser, enabling the user to further investigate the content. This is enabled with the help of the sentence functionality introduced in Section 3.3.

4.2.3 *Document Browser.*

The document browser provides on overview of the co-occurrences related to a graph. The user can browse all instances by firstly selecting the document, and secondly, going through all the co-occurrences in the document, which are then highlighted for her. If a source URL for a document is given, a link allows the user to inspect the original page. For a Wikipedia data set, as employed in our example, this can make sense if a user

[4] https://github.com/PhilipEHausner/python_heideltime

[5] https://visjs.org/ , accessed 19. June 2020



Enter Query Term: Yorktown, co

Select Granularity ☑ Year ☑ Month ☑ Day

Figure 3: A query for the term *Yorktown*. All dates for which the time-centric graph contains a node with the label *Yorktown* are highlighted. Further search terms can be selected from autocompletion suggestions.

wants to correct an error on the Wikipedia page she finds during investigation of the data. So far, documents are not re-ranked for specific nodes, but simply ordered alphabetically.

4.2.4 Query Interface. The query interface at the top provides the user with means to search for terms occurring across all time-centric graphs. After querying, the system highlights all dates for which a node in the associated graph carries the same label as the query term. It is also possible to query for multiple terms by specifying them as a comma-separated list. The system then only takes graphs into account that contain all the labels. Querying itself is engineered efficiently by constructing a hash index that maps each label to all graphs in which it appears. Additionally, the user is provided with query suggestions of terms contained in the graphs when she starts to type in the query field, based on case-insensitive string matching. Suggestions are sorted in descending order of the sum of the node weights across all graphs.

4.3 User Scenarios

The interactive interface enables multiple common user scenarios, depending on the user's information needs. If she is interested in the temporal order of events, or the general time span relevant to a document collection, a top-down investigation is enabled by starting from the zoomed out timeline. Specifically, drilling down from a coarse date granularity (e.g., year) allows for a very general overview and identification of key entities. Since other dates are also part of the graphs, it can also hint to related events that are closely tied to the scope the user currently focuses on. Using the sentence functionality the user can then explore the underlying data and investigate details of the events of a specific date. In the process, the user potentially finds entities that she is interested in and wants to find out more about. In such cases, she can query the data set for this term and investigate dates where the entity was also mentioned. Hence, the system does not provide one streamlined way to absorb knowledge, but involves the user in the exploration process that often needs a back-and-forth between the different elements of the interface. This is further enhanced by the date aggregation of documents, which bundles relevant parts together. Instead of reading potentially thousands of documents, all crucial information is in one place utilizing a concise timeline representation. On the other hand, the sentence functionality still allows to access the source documents in a systematic time-centric approach. A major secondary need is the the question about *specifics* of a certain person at a certain date. Starting with prior knowledge of some portion of the data set, the query interface allows a user to also explore in a bottom-up fashion, quickly identifying relevant parts of the document collection, as well as the main time frame during which an event occurred. Coupling this with co-occurrence based search further extends the capabilities of this approach. Alternatively, a starting point from a specific day gives similar insights of a narrow portion of the data set, and also allows for a quick expansion of the context into the broader frame of reference.

4.4 Data Set and Online Demonstration

For demonstration purposes, we extracted a document collection from the English Wikipedia related to the American Civil War that was fought between 1861 and 1865. The data set consists of the main article about the Civil War[6] as well as all English Wikipedia articles linking to this page. The articles are then split into subsections and all documents whose title are either a date, or contain expressions as *list of*, *references*, *external links*, and the like, are removed. This subdivision is beneficial in certain processing steps, since HeidelTime as well as the extraction of time-centric co-occurrences performs better if only smaller sections are taken into account. For graph generation, a window size of $x = 2$ is used, which provides a reasonable trade-off between graph size and context sensibility. The final data set consists of 211,672 articles resulting in 2,221 individual graphs between 1860 and 1865 that are used for the demonstration. The data set as well as the underlying documents can be explored using the proposed system in our online demonstration (see the first footnote). In the demonstration itself, all graphs are reduced to the 25 highest-ranking nodes with regard to the tf-itf score to minimize computation costs for users. This results in a total of 60,521 (indistinct) nodes and 339,004 edges across all graphs.

REFERENCES

[1] Omar Alonso, Ricardo Baeza-Yates, and Michael Gertz. 2007. Exploratory Search using Timelines. In *SIGCHI 2007 Workshop on Exploratory Search and HCI Workshop*, Vol. 1. 1–4.

[2] Ricardo Campos, Gaël Dias, Alípio Mário Jorge, and Adam Jatowt. 2014. Survey of Temporal Information Retrieval and Related Applications. *ACM Comput. Surv.* 47, 2 (2014), 15:1–15:41. https://doi.org/10.1145/2619088

[3] Philip Hausner, Dennis Aumiller, and Michael Gertz. 2020. Time-centric Exploration of Court Documents. In *Proceedings of Text2Story - Third Workshop on Narrative Extraction From Texts co-located with 42nd European Conference on Information Retrieval (CEUR Workshop Proceedings, Vol. 2593).* 31–37.

[4] Matthew Honnibal and Ines Montani. [n.d.]. Spacy: Industrial-Strength Natural Language Processing, version 2.1.8, https://spacy.io/, accessed 17. March 2020.

[5] Chae-Gyun Lim, Young-Seob Jeong, and Ho-Jin Choi. 2019. Survey of Temporal Information Extraction. *JIPS* 15, 4 (2019), 931–956.

[6] Jakub Piskorski, Vanni Zavarella, Martin Atkinson, and Marco Verile. 2020. Timelines: Entity-centric Event Extraction from Online News. In *Proceedings of Text2Story - Third Workshop on Narrative Extraction From Texts co-located with 42nd European Conference on Information Retrieval (CEUR Workshop Proceedings, Vol. 2593).* 105–114.

[7] Natalia Prytkova, Marc Spaniol, and Gerhard Weikum. 2012. Predicting the Evolution of Taxonomy Restructuring in Collective Web Catalogues. In *Proceedings of the 15th International Workshop on the Web and Databases 2012, WebDB 2012, Scottsdale, AZ, USA, May 20, 2012.* 49–54.

[8] Andreas Spitz, Satya Almasian, and Michael Gertz. 2017. EVELIN: Exploration of Event and Entity Links in Implicit Networks. In *WWW Companion.*

[9] Andreas Spitz, Jannik Strötgen, Thomas Bögel, and Michael Gertz. 2015. Terms in Time and Times in Context: A Graph-based Term-Time Ranking Model. In *Proceedings of the 24th International Conference on World Wide Web Companion, WWW 2015, Companion Volume.* 1375–1380.

[10] Julius Steen and Katja Markert. 2019. Abstractive Timeline Summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization.* 21–31.

[11] Jannik Strötgen and Michael Gertz. 2013. Multilingual and Cross-domain Temporal Tagging. *Language Resources and Evaluation* 47, 2 (2013), 269–298.

[6] https://en.wikipedia.org/wiki/American_Civil_War, accessed on 19. June 2020