Omar Rogelio Alonso
August 2008
Computer Science

Temporal Information Retrieval

Abstract

Time is an important dimension of any information space and can be very useful in information retrieval tasks such as document exploration, similarity search, and clustering. As search applications keep gathering new and diverse information sources, presenting relevant information anchored in time becomes more important for exploration purposes.

Temporal information is available in every document either explicitly, e.g., in the form of temporal expressions, or implicitly in the form of metadata. Recognizing such information and exploiting it for document retrieval and presentation purposes are important features that can significantly improve the functionality of search applications.

In this research, we introduce a temporal document analysis framework for analyzing document collections from a temporal perspective in support of diverse information retrieval and exploration tasks. Our analysis is not based on document creation and/or modification timestamps but on extracting time from the content itself. An examination of a document with an emphasis on time data can lead into interesting discoveries. We study models and techniques to inspect a document based on its temporal content that can serve as the basis for comparison, clustering, and categorization. A core part of the framework is a system for annotating documents with temporal information that is implemented using existing tools.

We present an add-on to traditional information retrieval applications in

which we exploit various temporal information associated with documents to present and cluster documents along timelines. Using temporal entity extraction techniques, we show how temporal expressions are made explicit and used in the construction of multiple-granularity timelines. We discuss how hit-list based search results can be clustered according to temporal aspects, anchored in the constructed timelines, and how time-based document clusters can be used to explore search results.

We examine how temporal information can be used for ranking purposes. We also propose an alternative document snippet technique that leverages new time aggregated measures that can help to highlight the most salient events.

Finally, we present Pacha, an exploratory search system that combines the methods introduced earlier for exploring search results in timelines. Pacha is one of the many applications that can be developed using the framework.

_____

Professor Michael Gertz
Dissertation Committee Chair

Temporal Information Retrieval

By

OMAR ROGELIO ALONSO
B.Sc. (Universidad Nacional del Centro 1994)
M.Sc. (Virginia Tech 1997)

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Computer Science

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

_____

_____

_____

Committee in charge

2008

To Julia, Valentina, and Lucas

Acknowledgments

I have been very fortunate in having tremendous support while doing this research. First and foremost, I want to thank my two advisors Dr. Michael Gertz and Dr. Ricardo Baeza-Yates for their constant support and encouragement during this thesis. Michael for his patience to work with an industry guy who has strong opinions (that is me) and to go above and beyond the call of duty to solve any administrative issues with Davis. Ricardo for making the time in his busy schedule and always willing to meet at anytime, anywhere to discuss a topic and provide feedback. This thesis was developed in a truly "virtual lab" way.

I want to thank Dr. Prem Devanbu for his constant support and advising while I was considering the idea of pursuing a PhD. Dr. Vladimir Filkov for the valuable feedback on early versions of this research. Wei Yu for her assistance in histogram representations and for answering many questions about time series.

I would like to thank my family. During this long journey, your love, support, guidance, and endless patience have been truly inspirational.

This thesis was in part possible by the generous support of my former employers via the tuition reimbursement program. Most of this research was assisted financially by Oracle Corp. and SAP Research. In particular, a big thanks to my managers at Oracle for allowing me to start in the program.

Last, but not least, I thank my friends and colleagues (too many to list here) for providing support and friendship that I needed.

Contents

List of Figures

List of Tables

Abstract

Time is an important dimension of any information space and can be very useful in information retrieval tasks such as document exploration, similarity search, and clustering. As search applications keep gathering new and diverse information sources, presenting relevant information anchored in time becomes more important for exploration purposes.

Temporal information is available in every document either explicitly, e.g., in the form of temporal expressions, or implicitly in the form of metadata. Recognizing such information and exploiting it for document retrieval and presentation purposes are important features that can significantly improve the functionality of search applications.

In this research, we introduce a temporal document analysis framework for analyzing document collections from a temporal perspective in support of diverse information retrieval and exploration tasks. Our analysis is not based on document creation and/or modification timestamps but on extracting time from the content itself. An examination of a document with an emphasis on time data can lead into interesting discoveries. We study models and techniques to inspect a document based on its temporal content that can serve as the basis for comparison, clustering, and categorization. A core part of the framework is a system for annotating documents with temporal information that is implemented using existing tools.

We present an add-on to traditional information retrieval applications in which we exploit various temporal information associated with documents to present and cluster documents along timelines. Using temporal entity extraction techniques, we show how temporal expressions are made explicit and used in the construction of multiple-granularity timelines. We discuss how hit-list based search results can be clustered according to temporal aspects, anchored in the constructed timelines, and how time-based document clusters can be used to explore search results.

We examine how temporal information can be used for ranking purposes. We also propose an alternative document snippet technique that leverages new time aggregated measures that can help to highlight the most salient events.

Finally, we present Pacha, an exploratory search system that combines the methods introduced earlier for exploring search results in timelines. Pacha is one of the many applications that can be developed using the framework.

Professor Michael Gertz
Dissertation Committee Chair

Chapter 1

Introduction

Search is a fundamental activity in today's modern world. Every day millions of people search for any kind of information on the Internet, enterprise environments, or on their own local computers. People seek information because they have a *need* or *goal* to satisfy, and a search system is a tool that can help achieve what they have in mind. A goal or *information need* has a wide range, from finding a car mechanic nearby, staying informed about a potential business acquisition, learning about a new vaccine, to explore a history piece in the last century, and so on.

One of the most preferred information access tools is a *search engine*. A search engine is a type of *information retrieval* system designed to help find information on the Web, inside a corporate network, or on a personal computer.

From the service's perspective, a search engine allows a user to enter a *query*, which can be a formulation of a goal in an input box, and presents a list of sorted search results, typically in the form of a list of pointers to documents. The result set contains a number of information items, usually around ten, which allow the user to quickly scan them. The level of detail for each item differs among search engines, but one can identify commonalities like the Web page title, URL (Uniform Resource Locator), a short snippet or text summary with highlighted query terms, and some metadata like the file size or date.

The default criterion for arranging search results is to order them by their *relevance*

to the query. The search engine applies algorithms to compute metrics that represent how close the answer is to the query. When a result set is precise and its presentation in a user interface is appealing, then the system is doing its best to satisfy the user's information need. Ultimately, relevance is the user's perception of the quality of the results after a visual inspection.

Once the results are presented, the user looks at them sequentially taking cues from the user interface, like the title or snippet, before selecting a document or Web page and looking at its content. There is little room for more exploration besides the appropriate selection and visual lookup. In some systems there is the ability to sort a search result by other metadata like date, with relevance being the default. Still, this exploration feature is very limited given the usefulness of time information.

Time is ubiquitous in any system. The value and cost of information have a direct relationship with its temporal characteristics. In some cases, time is considered one of the most important factors when deciding if an arbitrary object is obsolete or new.

Success through time is also a useful metric to see the importance of material things, concepts, and ideas over time. In our daily life we make observations and decisions based on time. Mundane expressions like *new*, *old*, or *classic* carry temporal information. After all, time has been studied since ancient times and it is part of all human life and activities.

Time is also an important dimension of any information space and can be very useful in information retrieval. Current information retrieval systems and applications do not take advantage of all the time information available in the content of documents to provide better search results and user experience.

1.1 The Role of Time in Information Retrieval

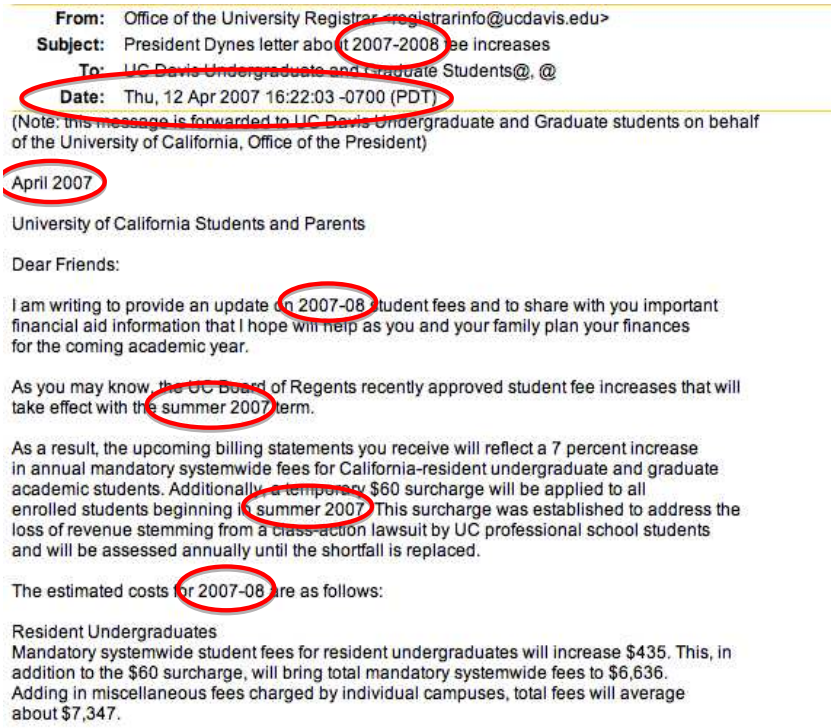
As the amount of generated information increases so rapidly in the digital world, the notion of using time as another factor becomes more important for indexing, organizing and retrieving content.

Time has been studied in other information technology areas like temporal databases, data mining, topic detection and tracking, information extraction, question-answering and summarization [67]. Time and time measurements can help in recreating a particular historical period or set the context of a document. As an alternative to ranking techniques like those based on popularity (e.g., [17, 69]), time can be valuable for placing search results in a *timeline* for exploration purposes. Recently, time is also placing a central role in some of the Semantic Web efforts [51, 83].

A quick look at any of the current search engines and information retrieval systems shows that the temporal viewpoint is restricted to sorting the search results hit list by date only. The date attribute is mainly the creation or last modified date of a Web page or document. In some cases it can be misleading since the timestamp provided by a Web server or any other system may not be accurate. Other search applications provide a range date search as part of the advanced search options. Still, the search results are filtered again by a date attribute. For searching purposes, the time axis is mainly based on document metadata. Recently, Google has added the `view:timeline` mechanism that allows to see certain search results presented in a timeline [99].

On a different scenario for using time information, the Wayback Machine is a digital time capsule created by the Internet Archive that allows users to see archived versions of Web pages [13]. This feature is for browsing purposes only and does not include search functionality.

Time information in a document can be found in several ways besides metadata. Because a document is the main object that any information retrieval application has to deal with, it is important to identify specific temporal information. A document contains a limited number of words that are separated by a blank space or any other punctuation mark delimiters. Words or combinations of them can form *temporal expressions* like "yesterday" or "October 2006". Usually every document has some sort of temporal expressions. We believe that extracting those expressions is a very important step for adding a time dimension to information retrieval.



From: Office of the University Registrar <registrarinfo@ucdavis.edu>
Subject: President Dynes letter about 2007-2008 fee increases
To: UC Davis Undergraduate and Graduate Students@, @
Date: Thu, 12 Apr 2007 16:22:03 -0700 (PDT)

(Note: this message is forwarded to UC Davis Undergraduate and Graduate students on behalf of the University of California, Office of the President)

April 2007

University of California Students and Parents

Dear Friends:

I am writing to provide an update on 2007-08 student fees and to share with you important financial aid information that I hope will help as you and your family plan your finances for the coming academic year.

As you may know, the UC Board of Regents recently approved student fee increases that will take effect with the summer 2007 term.

As a result, the upcoming billing statements you receive will reflect a 7 percent increase in annual mandatory systemwide fees for California-resident undergraduate and graduate academic students. Additionally, a temporary $60 surcharge will be applied to all enrolled students beginning in summer 2007. This surcharge was established to address the loss of revenue stemming from a class-action lawsuit by UC professional school students and will be assessed annually until the shortfall is replaced.

The estimated costs for 2007-08 are as follows:

Resident Undergraduates
Mandatory systemwide student fees for resident undergraduates will increase $435. This, in addition to the $60 surcharge, will bring total mandatory systemwide fees to $6,636. Adding in miscellaneous fees charged by individual campuses, total fees will average about $7,347.

Figure 1.1: UC Email message.

As examples lets take a look at two different documents. The first one, shown in Figure 1.1, is an email about new student fees at the University of California. As one can see, the timestamp of the message is represented as the `Date` attribute and is provided by the system. Looking at the body, one can identify more temporal information like the month of the email (April), an entry on the academic calendar (Summer 2007), and a time period (2007-2008). Temporal expressions are highlighted with an oval in the figure.

The second example, shown in Figure 1.2, is an excerpt from a long and detailed article adapted from the BBC Website about political news. Once again, we see a very precise timestamp attribute, in this case the last modification date for the news wire. The news article contains several references to the past (1997) and the future (27 June).

As one can see from the examples, the content of a document has temporal infor-

```
Last Updated: Thursday, 10 May 2007, 17:30 GMT 18:30 UK

Blair will stand down on 27 June

Tony Blair has announced he will stand down as prime minister on 27 June.

He made the announcement in a speech to party activists in his Sedgefield
constituency, after earlier briefing the Cabinet on his plans.
...
He said expectations had probably been "too high" in 1997, but he defended
his government's record in office.
```

Figure 1.2: News fragment. Adapted from the BBC website.

mation that is not necessarily captured in the metadata. Understanding the value of time in the document metadata and within the document is very important.

In our work, we are primarily interested in exploiting and presenting various time information embedded in documents as *relevance cues* to highlight an object (e.g., parts of a collection, a single document or parts thereof in some context) in an information seeking and retrieval process [10, 53, 88]. Interestingly, there has only been little work on fully utilizing the temporal information embedded in the text of documents for exploration and search purposes.

1.2 Motivating Examples

We can argue that for certain types of data sources or depending on a particular application, the role of time is implicit in the query. As notation, we use square brackets [ ] to mark the beginning and end of queries. So [“world cup”] means to do a phrase search for “world cup”, while [world cup] means just to type in those words without the quotes when issuing a search in the input box of search engine.

If one is looking for information about [Q3 earnings] for a company, it is assumed that the query means information that pertains to the third quarter of the current fiscal year. This is typical for content that has a short time span such as news articles. News

is essentially new information or current events. If the news is old, it is generally assumed that it is not relevant. On the other hand if the query is [tsunami disaster], and given that the event happened more than a year ago, one expects to retrieve the fact of the event, not necessarily the latest news (although that can be the case, too). A query for [football world cup] now returns information about the past event in Germany. But every football (soccer) fan knows that the event happens every four years. Another example is the query [Iraq war], where the results are based on the latest events with little content from the early 1990s war. In short, time does impact the quality of the search results. Ideally, we would like a search engine to be aware of the temporal information embedded in documents and present search results in a temporal context.

Contextual advertising algorithms display graphical or text-only ads that correspond to the search query terms. These ads tend to share a similar context as the user's search query. After all, an ad is basically a short fragment of text. Online advertisement can also benefit from a temporal analysis. For example, the query [costumes] may have two good seasonal matches like Halloween (October) and Carnival (February).

From a corporate perspective, enterprise search engines have to be able to retrieve all kinds of documents in any possible format. Based on the Sarbanes-Oxley legislation where public companies are required to keep records of certain assets (email for example) for periods of time, it is becoming important to store every reference to a document - even if the document is no longer present [90]. This has prompted a new market called *e-discovery*, which applies information retrieval and other search techniques for the legal domain and/or sensitive corporate data with the intent of using it as evidence in civil or criminal cases. In case of an audit, *forensic retrieval* (searching the past) can be performed to determine the existence of respective document from the index. A query like [sell stock before Q3 earnings] in a particular timeframe determines its relevancy. For the forensic retrieval user, the important scenario is to reconstruct relevant documents for a particular period of time.

Documents may also have temporal expressions about future events, in particular news. As presented in previous research [16], about 80% of the news about the future refer

to the immediate future (days, weeks, a few months). In opposite to forensic retrieval, *future retrieval* consists in extracting temporal information from news and combine it with standard full-text retrieval to answer queries that mix text and time.

There are other vertical search applications like medical informatics, where certain data sets such as patient discharge summaries contain very detailed time information. In reconstructing a patient's medical history, the ability to find events and present them in a timeline is a key aspect for its accuracy. In another example, a user might be interested in patients that had a similar chronology in the development of a disease. While it is easy to query for documents relevant to that disease, there is currently no means to explore the temporal similarity of the resulting documents.

From a user's perspective, the relevance of a query has a temporal aspect. The more data sources an information retrieval system acquires, the more important the temporal aspect can be in the retrieval process. Instead of assuming that the user wants relevant search results implicitly sorted by date, it would be interesting to investigate a system that is aware of time for relevancy and shows results in a temporal context. It can be useful to filter the "trendy stuff" from the rest.

Every retrieval engine combines search and browsing features by categories (predefined or generated). One can also extend the notion of browsing by adding temporal attributes. A clear advantage is that there is a predefined order for temporal items. For example, Wednesday is before Thursday and the year 2002 includes the range from January 1st to December 31st. All of us are used to use a calendar to see main events in time. A combination of calendar and browsing looks like an interesting alternative to search and navigate large document collections from a temporal perspective.

Finally, since the system can detect time events, it would be beneficial if it can make connections and therefore trace documents. The main point here is that time is crucial to all reasoning. It is very important to know what happened before what, so one can discuss cause and effect. In the example of the query [sell stock before Q1 earnings], the identification of stock transactions after that time frame would indicate an effect on the

query (the email that contained the message about selling prompted people to dump the stock). If we use the software engineering domain where one is looking for a document about a feature requirement, it would be useful if the system can trace if certain requirements have been met in a final product description or if a source code transaction was made after that timeframe.

In summary, temporal information embedded in documents in the form of temporal expressions provides an important means to further enhance current search engines. Recognizing such temporal information and exploiting it for document retrieval and presentation purposes are important features that can significantly improve the functionality of search and exploration applications.

1.3 Challenges and Objectives

In this thesis, we investigate how to identify temporal information in document collections and how to use them for search, retrieval, and exploration purposes. The underlying framework for such study consists of a number of temporal measures and operations that can be thought of as building blocks for developing temporally aware information retrieval applications.

Our work relies on extracting temporal information from documents and making it explicit in the form of temporal document profiles in a document annotation process. As part of that step, a temporal order that contains the distribution of temporal expressions in a document is also computed. The temporal document profiles are anchored in well-defined timelines that support different levels of temporal granularity. These profiles are used in combination with standard document ranking techniques to construct hit list clusters in which documents are grouped hierarchically based on their temporal annotations. The new timeline that contains all relevant content is presented as part of an exploratory search application. A temporal snippet that favors time information in sentences is also included.

The first objective is to design a document model that captures time information and makes it available in a standard representation. The challenge is to provide a model

that works for any document collection and that can represent all the information needed for clustering and exploration.

The correct identification of time is a very important part of the infrastructure work. This is not a trivial step due to the nature of the English (or any other) language. Using temporal entity extraction techniques based on natural language processing, we illustrate how temporal expressions can be made explicit as temporal document profiles. The challenge in this process is to distinguish from the output of the temporal extraction what expressions are useful for our work.

We study temporal characteristics of documents than can be useful for information retrieval and exploration tasks. We introduce a number of temporal measures like temporal specificity, coverage, and order among others that represents different aspects of a document from a time perspective. This can be useful to categorize if a document is a *résumé*, a news article, or a bibliographic page for example.

We apply those metrics to different document collections to get a better understanding about what and how temporal expressions appear in certain topic specific collections. This observation leads into the design of a new document snippet technique that focuses on the usage of time and events.

Based on temporal document profiles, we discuss how relevant search results can be organized according to diverse temporal aspects and how time-based document clusters can be used to explore search results. The objective of this part is to establish the main features that such clustering needs and its best representation. We introduce a time-based clustering algorithm, realized as add-on to information retrieval applications and provide a prototype that demonstrates its feasibility. To show experimental results of the many techniques introduced, we used different document collections.

We summarize the main findings of a user study that provided useful understanding of a wide range of potential scenarios for using temporal analysis. Some participants provided concise descriptions of information needs that showed the many facets of the usage of time for search.

Finally, we present how to use temporal information for the construction of new interfaces that combine search and browsing. Adding time as part of exploratory search tasks can lead to interesting discoveries. We propose to use well-defined timelines as an alternative view for presenting search results that have rich temporal expressions or where the usage of time plays an important role. We also take advantage of certain information visualization techniques to show time information as a document preview. The goal of this application is to combine the ground work presented earlier with interaction techniques that can be useful as part of a sense making task.

This dissertation presents the following main contributions, in particular:

(i) A detailed description of a temporal document annotation model, including a process to extract diverse types of temporal information from documents and representing such information in the form of temporal document profiles.

(ii) A temporal analysis study, including specific measures for analyzing a single document based on its temporal information. This is useful for document comparison and ranking purposes.

(iii) A time-based clustering algorithm called *TCluster* that extends standard document retrieval techniques by clustering result documents based on their temporal profiles. This algorithm is the core part for constructing and presenting search results along different timelines.

(iv) A temporal snippet algorithm called *TSnippet* that uses the proposed measures for extracting the most salient parts of the document.

(v) A prototype implementation that illustrates the realization of the underlying system architecture and document processing pipeline using standard software components.

(vi) An evaluation of the new approach using different document collections that demonstrates the feasibility and utility of the proposed approach.

(vii) An exploratory search application that uses time as one of the main examination features. This application also combines new and existing time-based visualization metaphors.

1.4 Structure of the Thesis

In Chapter 2, we present the necessary background in information retrieval, clustering, information extraction, natural language processing, and related applications. In Chapter 3, we establish the foundations of our current research including a temporal document model, temporal annotations, temporal document profile, and a mechanism for annotating document collections. A prototype implementation is also discussed along with its evaluation. In Chapter 4, we analyze a document from a temporal perspective by defining different temporal aggregate measures and a document similarity measure that can be used for ranking and exploration purposes. The last section contains case studies and their evaluation. Chapter 5 covers the time-based clustering algorithm, its implementation and evaluation. We also define and introduce a temporal snippet technique and how it can be used in search tasks. Chapter 6 details an application for exploratory search using timelines as the main model for retrieval. We also include a user study that provided valuable information where different search and exploration scenarios can be derived. Conclusions and future research trends are presented in Chapter 7.

Chapter 2

Background

The objective of this chapter is to provide the basic terminology, concepts, and models for information retrieval, document clustering, information extraction, and related applications.

We start with an overview of the main ideas behind modern information retrieval systems in Section 2.1. We then describe clustering in Section 2.2, a very important component of any search system. We continue with Section 2.3 on information extraction and natural language processing, which are important techniques of our approach. Finally in Section 2.4, we present related applications like exploratory search, temporal retrieval of news, and searching the future.

2.1 Information Retrieval

In this section we provide a summary of the main concepts and techniques in modern information retrieval systems and applications. There is a rich bibliography on this active field that includes classics [108], modern systems fundamentals ([17], [18], [44], and [69]), semantic and cognitive representations [40], and geometry-based modeling [113]. Each book provides a different angle to this fascinating and dynamic area opening new doors for research. A detailed survey on probabilistic models by Crestani *et al.* is presented in [30].

Information retrieval (IR) helps users find documents that satisfy their information

needs. At a glance, this looks pretty obvious but is not. Users are not very expressive in what they want. Also the information they provide can be ambiguous making the task even harder. A user will judge if a query satisfies an information need if the search results are *relevant*. An IR system processes documents or textual content that represents near 90% of the data that is stored in some sort of digital media. Examples of information retrieval systems are search engines like Google and Yahoo!.

A document is the unit of retrieval. It might be a paragraph, a section, a chapter, a Web page, an article, or a whole book. An interesting discussion about the meaning of documents in the digital age can be found in [63].

A document also has the following characteristics:

1. Language. Different cultures have different languages with their own particular set of rules. From a digital perspective, languages differ in character sets (Western languages, Chinese, Korean, Japanese, etc.) and other lexical features.

2. Syntax. The syntax of a document can express structure, presentation style and semantics. In many cases one or more of these elements are implicit or given together. A document typesetting language like TEXprovides rich mechanisms to control the layout and presentation of text.

3. Structure. With the arrival of markup languages like XML (Extensible Markup Language), it is possible to identify structure in documents and apply retrieval operations to certain parts of a document.

There are several models for information retrieval systems and we do not plan to provide a comprehensive survey of each here. At a high level we describe the models following the taxonomy provided in [17].

1. Classic.

   (a) Boolean: based on set theory and Boolean algebra.

   (b) Vector Space: uses weights to terms and provides a degree of similarity.

   (c) Probabilistic: captures the IR problem within a probabilistic framework.

2. Alternative Set.

   (a) Fuzzy Set Model: represents documents and queries through keywords that are partially related.

   (b) Extended Boolean: inclusion of vector space features.

3. Alternative Set Theoretic.

   (a) Latent Semantic Index: maps each document and query into a lower dimensional space which is associated with concepts.

   (b) Neural Networks: captures the IR problem within a neural network framework.

4. Alternative Probabilistic Models.

   (a) Bayesian networks: formalism that combines different source for ranking purposes.

   (b) Inference network: observation of a document is the cause for belief in the variables associated with its index terms.

   (c) Belief network: generalization of inference network.

5. Structure Models.

   (a) Proximal nodes: model that allows the definition of hierarchical indexing structures over the same document.

   (b) Non-overlapping lists: division of a document in non-overlapping text regions.

6. Browsing.

   (a) Flat: exploration of the document collection, which has a flat organization.

   (b) Hypertext: exploration of a document collection that has a hypertext model like the Web.

(c) Structure Guide: exploration of a document collection using a directory.

The vector space model is one of the oldest and most widely used model for *ad-hoc* retrieval. Documents and queries are represented in a high-dimensional space, in which each dimension of the space corresponds to a word in the document collection. The most relevant documents for a query are those vectors that are close to the vector representing the query. For a document d, its vector $d = (w_1, \ldots, w_n)$ represents the indexed terms w_i in d.

2.1.1 Relevance and Similarity Measures

Relevance is an essential concept in information retrieval. Surprisingly, given the amount of research in the field, there is no clear definition due to the inconsistent terminology used in the community. A clear and in-depth study on the many forms and meanings of relevancy is presented by Mizzaro in [75] and [76]. He presents a four dimensional space as a lattice that represents: 1) information resources, 2) representation of the user's problem, 3) time, and 4) components. Another extensive study on manifestations and effects of relevance is presented by Saracevic in two excellent articles in [88] and [89]. Recently, Borlund introduced relevance in the context of information retrieval evaluation [19].

We propose the following definition for relevance in the context of a search activity.

Definition 2.1 *Relevance is the significance that a user expects from the quality of the search results given an information need at certain point in time.*

We perform retrieval in the vector space model by *ranking* documents according to how similar they are to the query. This similarity measure is usually the *cosine* similarity.

Definition 2.2 *Given a document d and a query q, the cosine similarity is defined as follows:*

$$\cos(d, q) = \frac{d \cdot q}{\|d\|\|q\|} = \frac{\sum_{i=1}^{n} q_i d_i}{\sqrt{\sum_{i=1}^{n} {q_i}^2}\sqrt{\sum_{i=1}^{n} {d_i}^2}}$$

A document contains hundreds of words so it makes sense to count and weight them in the vector space model. Term frequency, $tf_{i,j}$, is the number of occurrences of w_i in d_j. Document frequency, df_i, is the number of documents in the collection w_i occurs in. Collection frequency, cf_i, is the total number of occurrences of w_i in the collection or *corpora*.

Intuitively, term frequency means how salient a word is within a document. The higher the frequency, the more likely that the term is a good description of the document. Document frequency is how informative the term is in the collection.

A popular document frequency weighting scheme or *scoring* formula is *tf-idf*, which stands for term frequency and inverse document frequency respectively. Inverse document frequency (idf) measures the rarity of a term. The equation multiplies term frequency of a term in a document by the term's inverse document frequency.

$$idf_t = \log \frac{N}{n_t}$$

where N is the number of documents and n_t is the number of documents that contain t. The weight of a term t in a document vector d is then

$$w_{t,d} = tf_{t,d} * idf_{t,d}$$

This scoring mechanism basically tell us two important facts about a particular term. The more often a term appears in a document, the more important it is. The more often it appears in the entire collection, the less important it is since it is common to all documents.

2.1.2 Index Creation

A very important component of any information retrieval system is the indexing mechanism. The best way to describe the index creation is like a pipeline where the input is a collection of documents and the output the final index.

The first step is the *tokenizer*, which does perform lexical analysis and separates each word properly. The following step is to apply *linguistic models* for normalizing the tokens. For example, conversion to lowercases, punctuation, stemming, etc. Finally, the index called *inverted index* is created. An inverted index file contains a sorted list of records where each record includes a word and pointers to the documents where you can find that particular word. The list of records is usually called *dictionary*. The *posting list* contains pointers to all documents that contains a given token. Figure 2.1 shows a simplified version of the indexing pipeline for a short document d_1 that contains the text `University California Davis`. The posting list is sorted by document identifier.

With the inverted index data structure, composed of the dictionary and the posting list, one can retrieve documents that contain the term `California` as follows:

(i) Locate `California` in the dictionary.

(ii) Retrieve postings from posting list.

(iii) Present result set.

Adding a second document d_2 that contains the text `University of California Berkeley`, one can perform a boolean query [Davis and Berkeley] as follows:

(i) Locate `Davis` in the dictionary.

(ii) Retrieve postings from posting list.

(iii) Locate `Berkeley` in the dictionary.

(iv) Retrieve postings from posting list.

(v) Merge posting lists.

(vi) Present result set.

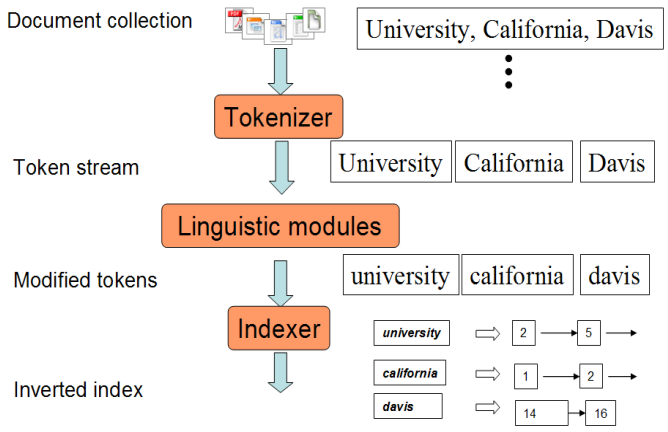


Figure 2.1: Indexing pipeline.

2.1.3 Evaluation

Measuring the quality of search results is a crucial aspect to determine how good the information retrieval application is. There are two widely use metrics for evaluating the outcome of a query: *precision* and *recall*.

Consider a set of relevant documents R and a query q. Given q, the system generates a set of documents A that satisfies the query. Let $\|Ra\|$ be the number of documents in the intersection of the sets R and A. Figure 2.2, adapted from [17], shows the interaction between the sets. We define recall and precision as follows.

Definition 2.3 *Given Ra and* R*, recall is defined as the fraction of relevant documents which have been retrieved.*

$$Recall = \frac{|Ra|}{|R|}$$

Definition 2.4 *Given Ra and* A*, precision is defined as the fraction of the retrieved documents that are relevant.*

$$Precision = \frac{|Ra|}{|A|}$$

The Text REtrieval Conference (TREC), co-sponsored by the National Institute of Standards and Technology (NIST) and U.S. Department of Defense, is a series of workshops that focus on several research aspects of IR [105].

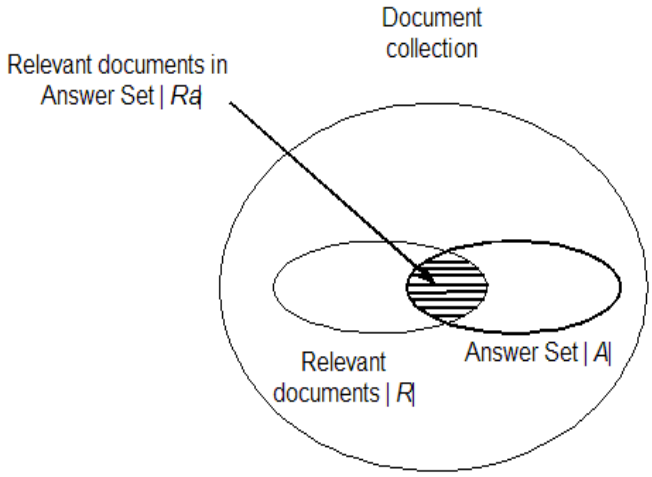


Figure 2.2: Precision and recall.

The main goal of TREC is to provide the infrastructure necessary for large-scale evaluation of text retrieval methodologies. A TREC workshop consists of a set *tracks*, areas of focus in which particular retrieval tasks are defined. One can think of this as data sets that are used as benchmarks for evaluating different aspects of IR systems.

Web retrieval is the application of IR concepts to the World-Wide Web, where the structure is mainly graph-based and has interesting properties. The PageRank, HITS, and similar popularity based algorithms employed by current Web search engines are such an example [17, 69]. Most of the research in this area concentrates on applying data mining techniques for search and retrieval. A good source that covers the latest trends is the book by Chakrabarti [25]. There is also a good survey by Kobayashi and Takeda on information retrieval and the Web in [60].

2.2 Clustering

Contrary to classification, the task of assigning objects to one or more *categories*, clustering is the unsupervised classification of patterns. In terms of information retrieval, clustering aims at capturing similarities or differences among items of a document collection so it can help the retrieval process.

The cluster hypothesis states "closely associated documents tend to be relevant to the same request" [108]. In other words, if there is a document from a cluster that is relevant to a query, then it is likely the other documents from that same cluster are also relevant. This is possible because clustering groups together documents that have words in common, so it is expected that these documents behave similarly with respect to relevance.

A very important aspect in a clustering algorithm is the similarity measure. In document clustering, the similarity measure is usually defined in the form of vector space similarity or distance as described earlier. Clustering is a very active area of research, so we only present a high level overview of the main approaches. Most of the information retrieval books already mentioned discuss clustering in detail. An excellent survey of clustering is given by Jain *et al.* in [55].

2.2.1 Flat Clustering

This class of algorithms creates a flat set of clusters without any explicit structure that would relate clusters to each other. We adopt the definition from Manning and Schütze [69] for flat clustering. Given a set of documents $D = \{d_1, \ldots, d_n\}$, a number of clusters K, and an objective function that evaluates the quality of a clustering, one wants to compute an assignment $\gamma\colon D \rightarrow \{1, \ldots, K\}$ that minimizes the objective function.

Probably the most widely used algorithm in this class is k-means. The objective of k-means is to minimize the average squared distance of documents from their cluster centers where a cluster center is defined as the mean $\vec{\mu}$ in a cluster w:

$$\vec{\mu}(w) = \frac{1}{\|w\|} \sum_{\vec{x} \in w} \vec{x}$$

Another well known clustering algorithm is Expectation Maximization (EM), which assumes data generated by a model and then tries to recover the original model from the data. This model-based clustering defines clusters and their document assignment.

2.2.2 Hierarchical Clustering

Hierarchical clustering outputs a hierarchy of clusters that is more informative than the set of clusters in flat clustering. A hierarchy is defined as a binary tree with N leaves, where N is the number of documents. There is no need to pre-specify the number of clusters for this type of algorithms. Examples of hierarchical clustering algorithms are:

(i) Agglomerative: bottom-up clustering merges clusters into larger units.

(ii) Single-link: the similarity of two clusters based on the similarity of their most similar members.

(iii) Complete-link: the similarity of two clusters is the similarity of their most dissimilar members.

(iv) Group-average agglomerative: evaluates cluster quality based on all similarities between documents in vector space.

2.2.3 Labeling

Labeling is a very important feature in clustering construction. A *label* is a valid description of the clusters that presents to the user a good portrayal of the content. This is a difficult problem because, in the case of documents, the labels have to be automatically generated and, at the same time, have to be human readable. The techniques for label selection range from term distribution comparison to named-entity extraction [69].

2.2.4 Evaluation

One way of evaluating a cluster output is to use a metric to determine the quality. An example is to measure the high intra-cluster similarity (documents within a cluster are similar) and low inter cluster similarity (documents from different clusters are dissimilar).

As replacement for user judgments, one can use a set of classes in an evaluation benchmark that is produced by human judges. One can then compute an external criterion that evaluates how well the clustering matches the benchmark.

2.2.5 Hit-list Clustering

Hit list clustering has emerged as an alternative mechanism to present similar documents without requiring the user to go through hundreds of items (see, e.g., [115, 110, 104]). Clustering of result documents can lead to better user interfaces and, therefore, to an improved user experience. As a recent study on user search experience shows, users do prefer to use clustering when they are trying to get an overview of or explore a topic [15].

Hit list clustering groups search results into categories that are derived from the actual search. Instead of processing the entire document set, hit list clustering uses a small document set that fits several well-known substring searching algorithms. Current hit list clustering engines like Vivsimo's Clusty rely on a separate search engine that provides some information like Web page title, URL, and document snippets for the construction of the clusters [115, 110].

2.3 Information Extraction

Entity extraction (also named-entity extraction) is the recognition of entity names (people and organizations), places, temporal expressions, and types of numerical expressions (currencies, measures, etc.) from data.

One can think of entity extraction as a function that, given data as input, produces a set of value pairs that consist of named entities and their associated types. For example:

```
Oracle Corp., COMPANY
John Smith, PERSON
Canada, COUNTRY
$100, CURRENCY
```

Using the UC email example (Figure 1.1), an entity extractor program should be able to identify the University as an organization:

```
As you may know, the <ENAMEX TYPE="ORGANIZATION">UC Board of
Regents</ENAMEX> recently approved student fee increases ...
```

Entity extraction is a key component of *information extraction* (IE). The goal of IE is to identify instances of a particular pre-specified class of entities, events and relationships in textual documents. It also includes the extraction of the relevant arguments of the identified events or relationships. The process of extracting data involves locating names and finding linguistic relations between them and other entities. The templates generated may be incomplete, but incomplete information is still better than no information at all.

There are two main approaches for building IE systems: knowledge engineering and learning. The knowledge engineering approach consists of a human expert who inspects the document collection and creates the rules that define those entities (e.g., Bill is a person name) [1, 26]. The rule creation and testing is done iteratively until no significant improvements can be made. The level of expertise is a crucial factor in the overall quality of the system.

The learning approach involves machine-learning techniques like Hidden Markov Models (HMM), decision trees, and maximum entropy modeling [29, 72]. Like any learning technique, the larger the training data the better the results.

There are a number of research projects that leverage IE as components for a wide range of purposes. The research in the category is either from the pure IE or Natural Language Processing side where they measure the quality of the extractor, to the more practical approach where IE is just a function.

A well-known information extraction system is GATE/ANNIE [31], which is currently used in many commercial and research applications. Other examples are LingPipe, a

suite of Java libraries for the linguistic analysis [64] and ThingFinder, a commercial multilingual information extraction SDK (Software Development Kit) [97]. A good introduction to information extraction and natural language processing is the book by Jackson and Moulinier [54].

2.3.1 Natural Language Processing

Natural Language Processing (NLP) studies the problem of semantically understanding natural human languages. In the last few years there has been significant work on the *statistical* aspect, which uses quantitative approaches to automated language processing. An in-depth treatment of the fundamental concepts and techniques from a statistical natural language perspective is the book by Manning and Schütze [70].

In this subsection we present some basic linguistic terminology and only describe the most relevant concepts to our research: sentence identification and part-of-speech tagging.

Linguists group the words of a language into classes that show similar syntactic behavior. These word classes are often called *grammatical categories*, also known as *parts of speech* (POS). In traditional English grammar, there are eight parts of speech: the verb, the noun, the pronoun, the adjective, the adverb, the preposition, the conjunction, and the interjection. The various parts of speech for a word are part of a *dictionary* or *lexicon*. A typical POS program distinguish these eight categories, but for more precise classification of word classes, there are well known sets of abbreviations, called POS *tags* for naming these classes.

A *sentence* is a string of words satisfying the grammatical rules of a language. The most used heuristic for detecting a sentence is to look for a number of words that ends with a ".". "?', or "!". There are exceptions to this rule, but it satisfies most of the cases.

The main goal of an NLP tool is to parse and understand human language. This has been a very difficult problem to solve since the field started. A more practical problem is to identify the structure in language without requiring complete understanding. *Tagging*

is the process that labels (tags) each word in a sentence whith its appropriate part of speech. The following example is adapted from [70]. Consider a sentence `The representative put chairs on the table.`, the POS program would tag it as:

`The <AT> representative <NN> put <VBD> chairs <NNS> on <IN> the <AT> table <NN>.`

Here the tags, which are represented in `<>`, follow the Brown/Penn convention that specifies for example: article (`AT`), preposition (`IN`), verb, past tense (`VBD`), plural noun (`NNS`), and singular or mass noun (`NN`) [70]. There is no standard for tagging so there are several conventions available. Tagging can be used in information extraction so it is not a coincidence that Markov models are well suited for this task.

2.4 Related Applications and Technologies

Large commercial search engines like Google or Yahoo! provide a very good experience for end-consumers looking for any kind of information on the Web. Their underlying model is based on Web retrieval techniques that perform extensive data mining techniques on search query logs and user click-through. Still, there are several search scenarios where they do not produce relevant content.

In this section we describe other related applications and those scenarios that are relevant to our work and not well covered by current Web search engines.

2.4.1 Searching the Future

The idea behind searching the future is to query current news articles about future events that have a high probability of occurring [16]. Instead of just searching for current/past events, this proposed new model is about searching the future. The approach is based on entity extraction for identification of temporal attributes and a combination with traditional ranking algorithms.

2.4.2 Temporal Retrieval with News Archives

The news data source seems to be very popular for every research project that involves adding some sort of time attribute to the retrieval aspect.

One of the earliest works has been the Time Frames project [61]. The idea here is to augment news articles by extracting time information using entity extraction approaches and then providing alternative visualization to see pattern of interests in certain topics. This paper also has a very good description of the importance of time for context disambiguation and as a reference mechanism.

Temporal patterns or trend detection is also a very active area. The idea is to identify main topics and present their activity over time to detect trends or, in certain cases, "fashion" topics that fade away later. Some researchers call this area topic detection and tracking (TDT) or temporal text mining (TTM) where the goal is to detect themes, construct a model that shows their evolution over time, and finally analyze their theme life cycle [5, 74].

Some projects have taken these ideas further. For example: analyze the development of a collection of documents over time to detect main topics and how these topics evolve [93]. This is a very interesting application of citation analysis and "burst" detection to the identification of the life cycle of a topic.

2.4.3 Exploratory Search Interfaces

A we stated before, online search has become an important activity for users at the office and at home. Internet search engines and other vertical search systems provide good services for those users who have a good understanding of their information needs or goals. However, as the production of digital content continues at a rapid phase, current search systems have limitations. When it is difficult to express a query due to lack of knowledge, users must navigate complex information spaces. This new search task requires browsing and exploration.

Users usually submit a tentative query with a few keywords and then react based

on the search results. Maybe they refine the query, take other things from there, identify cues, etc. This type of activity is called "exploratory search". Exploratory search can be seen as a specialization of information exploration. Exploratory Search Systems (ESS) have been developed to support serendipity, learning, and investigation, and generally allow users to browse available information [111].

2.4.4 Information Visualization

There is a common agreement that the amount of data nowadays has reached a point where any attempt to present information from a large amount of data in a simple tabular format fails. Even if the information was there, if it is not presented in a useful way, it can lead to uninformed decisions.

Information visualization is defined as "visual representations of abstract data to amplify cognition" [21]. In the context of large data sets, visualization can help users navigate as well as provide a summary of the data collection. Shneiderman summarizes the principles as the *visual information seeking mantra*: overview first, zoom and filter, then details on demand [94].

Information visualization tools have been around for many years yet they have not reached mass adoption. A possible reason is that without proper content organization and structure, there is not much room for visualization.

It is important to note that several of the research activities that include time have a visualization component. Providing a list of items sorted by time does not seem to be appealing since there are other metaphors like calendars or planners that capture timelines in a better way.

2.5 Summary

We presented the fundamental concepts of information retrieval and related topics. In particular, we outlined clustering and information extraction. We also described other relevant areas like natural language processing, exploratory search, temporal retrieval of

news, and searching the future. This background chapter serves as introductory content for the rest of our work.

Chapter 3

Time-Annotated Document Model

In this chapter, we establish the foundations of our temporal information retrieval approach. We study how time data appears in documents and how it can be extracted automatically. We introduce a time-annotated document model that serves as the central point for the rest of the thesis.

This chapter is organized as follows. The next section introduces the motivation and objectives. Section 3.2 presents a general introduction to time. Section 3.3 introduces how time information appears in documents. Section 3.4 presents our approach to time and events. Temporal expressions and their categorization are covered in Section 3.5. The concept of a document profile is presented in Section 3.6. An overview of temporal information extraction is covered in Section 3.7. A short introduction to the TimeML standard is presented in Section 3.8. The proposed temporal annotation pipeline and its evaluation are shown in Section 3.9 and 3.10 respectively. Finally, Section 3.11 presents related work.

3.1 Motivation and Objectives

We argue that there is quite a lot of temporal information in any document collection. In general, documents are created and consumed by humans, so for each type of document collection, there are particular characteristics that define the set in a clear way. For example, financial news (as in Figure 3.1 with highlighted time information) tends to

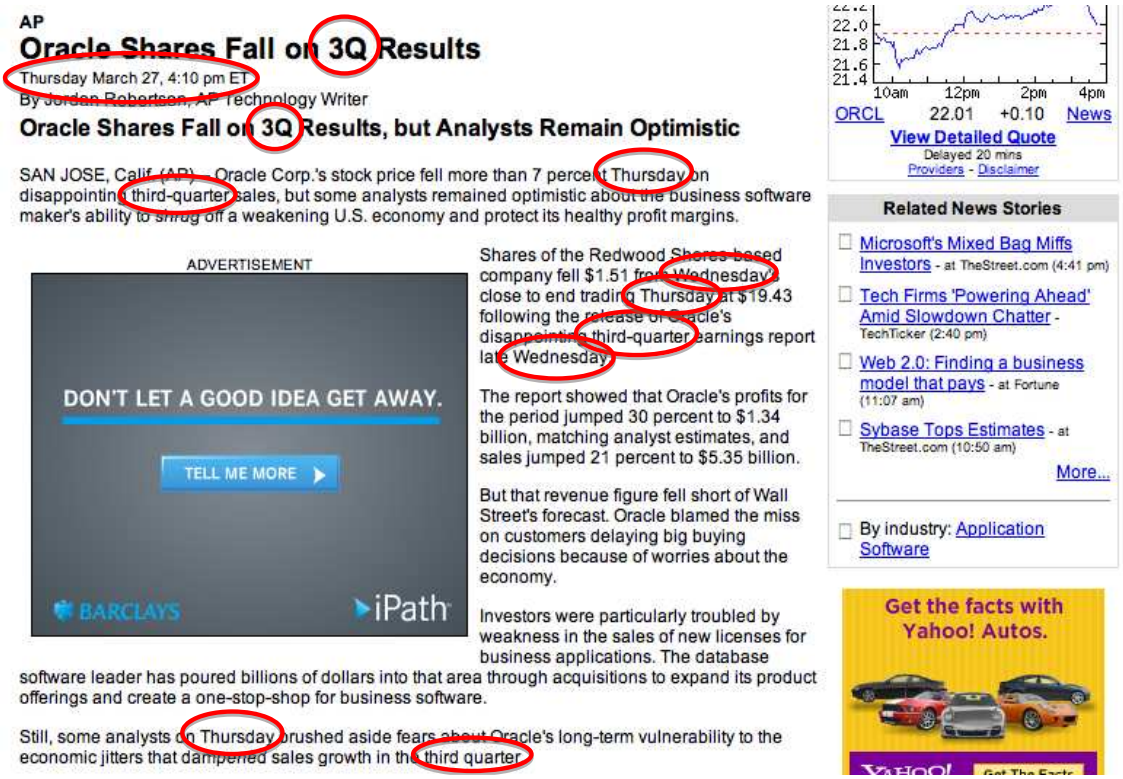


Figure 3.1: An example of a financial news article.

be rich in describing near future/past events; a résumé contains several references to the past in a very precise way; and project documentation involves phase milestones that are captured in time. These characteristics are independent of document format and language.

How can one take advantage of such time related information for document retrieval purposes that go beyond just sorting a hit list by a date attribute like document creation or modification time? How can we use temporal indications more effectively to search for documents of interest? How can we organize and present search results or document collections in a way that emphasizes temporal access?

To start addressing those questions we first need to establish a document model and provide a precise definition of what we mean by time in the context of searching and content organization.

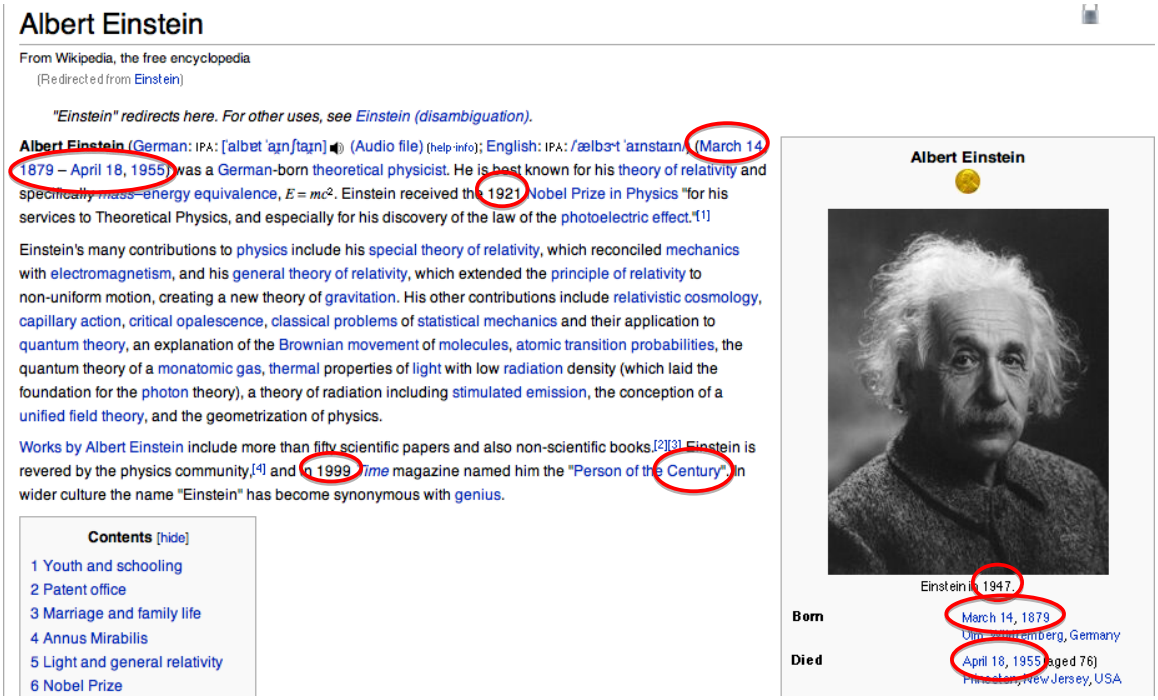


Figure 3.2: Albert Einstein information in Wikipedia.

Let us take a look at a Wikipedia entry about Albert Einstein as presented in Figure 3.2 [37]. When we open the document using a Web browser or making a copy of it, the Web server gives us the creation time or latest modification time. In case of Wikipedia, there are volunteers that edit the pages several times so a history of changes with timestamps is also available. This is metadata, like the footer on this particular entry: `This page was last modified 12:03, 1 June 2007`. By reading the document, we can identify some obvious time information like Einstein's date of birth ("March 14, 1879"), death ("April 18, 1955"), and certain events that happened in specific years such as his Nobel Prize ("1921"). This last part involves a more detailed analysis at the sentence and context level. Figure 3.2 has all *temporal information* highlighted as an example of such an analysis.

As one can see from the example, an important challenge is the automatic identification of all temporal information in a document. This step is language dependent and a precise recognition of every instance of time is crucial.

The next problem is then to organize all the time information extracted in a way that enables different levels of access and exploration.

To summarize, the objectives of this chapter are to:

(i) Develop the concept of a temporal document profile that represents those temporal expressions.

(ii) Develop and evaluate a process for annotating temporal documents automatically.

(iii) Precisely define the notion of time, events, and temporal expressions.

(iv) Review a number of existing approaches to extract time information from documents.

(v) Investigate the quality of temporal expressions.

3.2 What is Time?

Time has been a subject of study in many disciplines, particulary in philosophy, physics, logic, and art [107]. Since ancient times, philosophers have written at length about time and its impact on humankind, most notably by Immanuel Kant and Martin Heidegger. In physics, a practical usage of time is like a measure. For example, the time it takes an object to move from point A to point B. It is also a central part of the most advanced theories including Einstein's general relativity. Finally, in the arts time appears as a central object in paintings (Salvador Dalí), music (classic, pop), and literature (Jorge Luis Borges).

We start the study of time information in documents with a number of definitions to set up the right context. A look at an English dictionary gives the following definitions for "time" and "temporal":

Time: a) A non spatial continuum in which events occur in apparently irreversible succession from the past through the present to the future; b) An interval separating two points on this continuum; a duration: a long time since the last war; passed the time reading.

Temporal: of, relating to, or limited by time: a temporal dimension; temporal and spatial boundaries.

Since the notion of *event* appears quite often occurs in conjunction with the notion of time, its dictionary definition says:

Event: a) Something that takes place; an occurrence; b) A significant occurrence or happening; c) A social gathering or activity.

Time is defined operationally and involves the process of measuring the units chosen such as millisecond, minute, or century. A *timeline*, also known as a *chronology*, is a linear representation of events in the order in which they occurred. This sequence of related events is usually arranged in chronological order and presented in a line display drawn left to right or top to bottom (see Figure 3.3 for an example).

A *calendar* is a human designed system for representing physical time. A calendar defines the time values, called *granularities*, of interest to a user, usually over a specific segment of the timeline.

One calendar familiar to many is the Gregorian calendar, based on the rotation of the Earth on its axis and its revolution around the Sun. In the Gregorian calendar the granularities are year, month, day, hour, minute, and second.

In general, a calendar can measure time using a well-defined time unit. For example, an employee time card can be regarded as a calendar, which measures time in eight hour increments and is only defined for five days of each week.

The usage of calendar depends on the cultural, legal, and business orientation of the user. For example, business enterprises generally perform accounting relative to some fiscal year or quarters (four quarters in a year). However, the definition of a fiscal year varies depending on the type of company and its geographical location. Other examples are the academic calendar and sport seasons. No calendar is better than the other; the value of a particular calendar is determined by the application and the people that use it [95].

Figure 3.3 shows the timeline for the year 2006 and three points situated as follows: t_1 (October 1^{st}), t_2 (October 31^{st}), and t_3 (November 1^{st}). From the timeline, it is clear

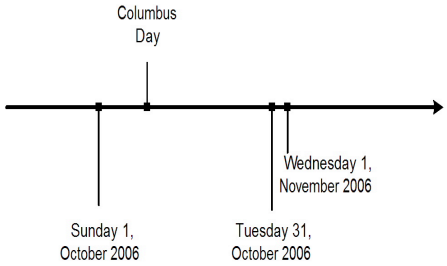


Figure 3.3: An example of a timeline for points and events.

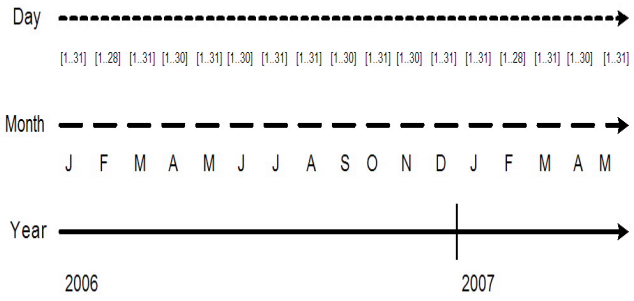


Figure 3.4: Mapping between granules.

that the precedence is $t_1 \prec t_2 \prec t_3$. The time period between t_1 and t_2 is the month of October in the Gregorian calendar. Columbus Day is an event that falls in this time period.

Figure 3.4 shows the mapping of time granularities between year, month, and day. Each year has twelve months and they are represented with first the letter of the month. Seven months have 31 days, four that have 30, and one that has 28.

When considering the issue of time in databases, we must distinguish between time as measured by the system and time as observed in the real world. The *valid time* for a fact is the set of time intervals during which the fact is true in the real world. The *transaction time* for a fact is the time interval during which the fact has been recorded in the database [80].

3.3 Time in Documents

Moving into the area of documents, things are very different because there are several ways of expressing time in a language like English. Without getting into a theoretical or applied scientific study of time in language, we provide an overview of the linguistics basics. A comprehensive study on language and time is given in the excellent book by Mani, Pustejovksy, and Gaizauskas [66].

The specific construction built into language for locating information in time is called *tense*. Tense is defined as the grammaticalized expression of location in time and involves marking, via change of form, of syntactic elements like verb and auxiliaries. For example, *"John ran a marathon"*.

Another language mechanism named *grammatical aspect* expresses whether an event is viewed as finished, completed, or ongoing. For example, *"John is building a house"*.

In corpora, an example of a temporal feature is temporal referring expressions that indicate times, durations or frequencies. Section 3.5 describes in greater detail the concept of temporal expressions. Examples of such times are: "on Friday", "March 21, 2007", "in the early 70s", "three weeks", and "monthly".

Since language is ambiguous and evolves over time, there is more than just one way of representing the same time information. "Christmas Day" and "December 25" mean the same but are expressed differently: from a holiday perspective and from an entry in a calendar. The same time event can have different entries in a calendar depending on cultures. For example, labor day in USA is different from the rest of the world. Finally, the same point in time (same index in a calendar) can be expressed in different languages: *Christmas* in English and *Navidad* in Spanish.

Going back to Einstein's page in Wikipedia and after reading the entire content, we can identify time information, draw a timeline and anchor events in his life even after his death (Figure 3.5). As we mentioned before, there is a granular aspect of physical time. In our case, the timeline was constructed around the year granule. The starting point is the date of birth and the current end point is the day of analyzing the document (time of

writing). Now, the document contains references to temporal structures that are aggregates of years like century or subcomponents like month. For that, the timeline has different levels of granules that go from the coarsest to the finest. Each highlighted time information points to a particular interval of the timeline. In other words, all temporal information has been *anchored*. A benefit of using a timeline is that it shows the density of the time granules. In this case, year references tend to be predominant. Distribution of time information per document is important and it will be discussed in detail in Chapter 4.

In order to anchor events in a timeline it is necessary to normlize time expressions to a common representation that can be mapped to a timeline. This is a necessary step to unify the different ways the same time can be expressed in English. For example: "3/21/2007", "21/03/2007", "March 21, 2007", and "the 21st of March in 2007".

The example also shows that, since the analysis of language is a complex matter, it is very important to identify all temporal information in documents as accurate as possible. Otherwise the anchoring would be incorrect.

3.4 Time and Events

So far we have presented a high level discussion on time modeled by units and the events that happen in time. In the following, we provide concise definitions and set up the context for the rest of our work.

A primitive and valid notion of time is a set of points without duration. This relation allow us to compare two points in time, which leads us to observe time as intervals or periods.

Definition 3.1 *The time domain $\mathcal{T}$ is a set of* equidistant *points ordered by a relation of precedence $\prec$. The range of values for $\mathcal{T}$ is* $[0, \infty)$.

Definition 3.2 *A time interval $[t_1, t_2]$ consists of two points t_1 and t_2, where t_1 denotes the start point, t_2 denotes the end point and $t_1 \prec t_2$.*

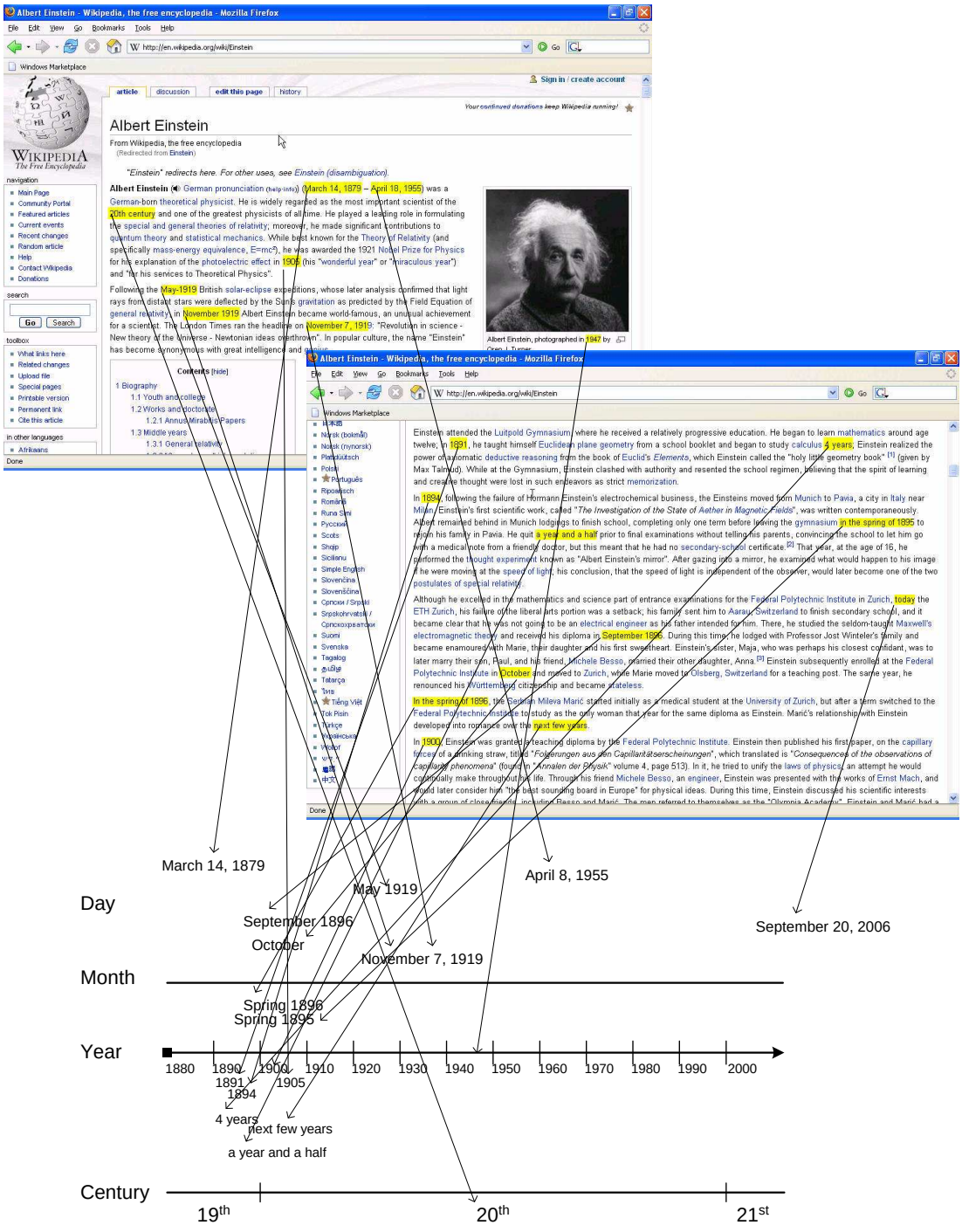


Figure 3.5: Timeline for Einstein's Wikipedia page.

An interval can be right open (left boundary provided, right undefined), left open (right boundary provided, left undefined), or closed (left and right boundaries are provided).

As the basis for anchoring documents in time, we assume a discrete representation of time based on the Gregorian Calendar, with a single day being an atomic time interval called *chronon*.

Definition 3.3 *A* base timeline*, denoted T_d, is an interval of consecutive day chronons.*

For example, the sequence "March 12, 2002; March 13, 2002; March 14, 2002" is a contiguous subsequence of chronons in T_d. Contiguous sequences of chronons can be grouped into larger units called *granules*, such as weeks, months, years, or decades. A grouping based on a granule provides us with a more coarse-grained timeline, such as T_w based on weeks or T_y based on years. Examples of week chronons in T_w are "3rd week of 2005" or "last week of 2006".

We assume the four timelines $\mathcal{T} = \{T_d, T_w, T_m, T_y\}$ for days, weeks, months, and years, respectively. The composition of granules naturally induces a lattice structure in $\mathcal{T}$. That is, we have the relationship $T_j \gg T_i$ if timeline T_j is (transitively) composed of granules of timeline T_i. In particular, we have $T_y \gg T_d$, $T_y \gg T_m$, $T_m \gg T_d$, and $T_w \gg T_d$, but not $T_m \gg T_w$ as months are composed of days and not weeks.

As indicated in Definition 3.1 associated with each timeline $T \in \mathcal{T}$ is a *precedence relationship* $\prec_T$ that allows to compare chronons. For two chronons $t_i, t_j \in T, t_i \neq t_j$, we then have either $t_i \prec_T t_j$ or $t_j \prec_T t_i$. For example, for the two day chronons $t_i =$ "March 12, 2004" and $t_j =$ "January 5, 2004", $t_j \prec_{T_d} t_i$ holds. This precedence relationship can easily be generalized to chronons from different timelines, e.g., "March 31, 2001" is after "April, 1999".

Note that a base timeline can be defined on any other atomic time interval, such as an hour (at a day) or a week. Also, any other calendar can be chosen, such as a financial calendar or academic calendar. These particular choices depend on the characteristics of the document collection to be time annotated and information exploration tasks, and do

not have any particular impact on the generality of the proposed approach.

Events should be anchored in time. For example, in Einstein's Wikipedia page, the following sentence denotes an important contribution of his life:

```
In 1905, while working in the patent office, Einstein published four
times in the Annalen der Physik}
```

Definition 3.4 *An event* e *is a phenomenon or occurrence located at time point* t_1 *in timeline* T.

We can think of an event as a chronon that has more semantics. In Einstein's Wikipedia page, the event is the publication of those four articles in that year.

The ordering of events is also important if we are interested in temporal relationships. Allen's interval algebra has been a core component of temporal reasoning and other applications that require relating intervals [6]. The list of all interval relationships in terms of X and Y is the following:

(i) X before Y

(ii) X equal Y

(iii) X meets Y

(iv) X overlaps Y

(v) X during Y

(vi) X starts Y

(vii) X finishes Y

All these different ways at looking at time and events can be very useful for document exploration purposes. For example, a user may express an information need in a time interval or would like to see search results present in a timeline where operations are available for further analysis.

3.5 Temporal Expressions

We now focus on the representation and extraction of temporal information from documents. As indicated in the introduction, a key aspect of our approach is to take a set of documents $\mathcal{D}$,

extract all types of temporal information associated with documents in $\mathcal{D}$, and make this information available to our analysis and exploration techniques.

The first type of such information is the *document metadata*, which appears as the date a document $d \in \mathcal{D}$ has been created or last modified. We denote such time related metadata of a document $d \in \mathcal{D}$ as *document timestamp* $d.ts$. Document timestamps typically can be obtained at document collection crawling or indexing time and can be anchored in the timeline T_d.

The second type of temporal information is a little bit more involved as it relates to the linguistic analysis of the textual content of documents. A suitable approach to identify time in text data is information or named-entity extraction, with *temporal entities* being time-related concepts. Such concepts are represented in the document text as (not necessarily contiguous) sequences of tokens or words. In particular, temporal entities can be made explicit in the form of a *temporal expression* that correspond to a chronon in some timeline. A temporal expression is a portion of textual content that express some direct or inferred temporal information. These expressions include mainly dates ("02/02/2002") or prepositional phrases containing time expressions ("on Monday").

Temporal expressions are recognized by an entity extraction approach using a time-based linguistic analysis. Expressions can be mapped to temporal entities and terms defined in some temporal ontology.

As a sample of the current research in the NLP community, there are two classifications of temporal expressions that are worth mentioning. The first one is from Schilder and Habel [91], where they distinguish *time-denotating* and *event-denotating* expressions in news articles. Our interest is in time-denotating expressions, which can be described as follows:

(i) Explicit reference: expressions like "02/02/2002" that are entries in a calendar system. It also covers time expressions like "3pm" or "Midnight" that denote a precise point in time.

(ii) Indexical reference: expressions that can be evaluated via an index time. For example, "today" or "next Sunday" must be evaluated with respect to the document timestamp.

(iii) Vague reference: expressions that indicate only vague temporal information and it is difficult to place them on a timeline. For example, "in several weeks", or "by Saturday the latest".

In the second classification, Pustejovsky *et al.* provide different ways in which time information is available in English [83]. This classification has a *question-answering* approach in mind, where the system is able to retrieve answers to questions posed in natural language. For example, to

answer a question about a person's current occupation, the system has to know all past occupations and select the newest one.

(i) Adverbial or prepositional phrases: expressions such as "in the 80s", "Monday afternoon", and "2/2/2004".

(ii) Indexical: expression that by themselves do not specify a specific time and need an indexical anchor like "next Friday".

(iii) Indeterminate: expressions that cannot be interpreted as part of a timeline, because their begin and end points are not clear. For example: "recently" and "in the Fall of this year".

(iv) Durations: expressions that refer to quantities of time such as "for 3 hours" and "a 2 hour flight".

(v) Anchored durations: expressions that specify the time of an event by making explicit the duration between the event and a time. For example: "two weeks from today".

(vi) Set of times: expressions that groups distinct parts of the timeline. For example: "every Thursday" and "2 weeks per month".

To summarize, depending on the application domain it may be necessary to provide a specific classification of how time information appears in text. Since we are educated consumers of NLP technology, we take a path closer to Schilder and Habel [91], and distinguish between explicit, implicit, and relative temporal expressions.

3.5.1 Explicit Temporal Expressions

Explicit temporal expressions describe chronons in some timeline, such as an exact date or year. For example, "December 2004" is an explicit expression that is anchored in the timeline T_m. Similarly, the expression "September 12, 2005" is anchored in T_d.

3.5.2 Implicit Temporal Expressions

Depending on the capabilities of the entity extraction approach and in particular its underlying time ontology, implicit temporal expression, such as names of holidays or events can be anchored in a timeline as well. For example, the token sequence "Columbus Day 2006" in the text of a document can be mapped to the expression "October 12, 2006", which is anchored in T_d. It

is also conceivable that a single temporal entity can be mapped to more than just one temporal expression, i.e., several chronons at once. For example, the token sequence "Winter season 2005" can be mapped to a combination of month, week, and day chronons in three different timelines. In general, implicit temporal expressions require that at least a year chronon appears in the context of a named event.

3.5.3 Relative Temporal Expressions

Relative temporal expressions represent temporal entities that can only be anchored in a timeline in reference to another explicit or implicit, already anchored temporal expression. That is, their anchoring depends on a chosen point of time reference or narration. For example, the expression "today" alone cannot be anchored in any timeline. However, it can be anchored if the document is known to have a creation date as a reference. Then it is likely that the expression can be mapped to that date. There are many instances of implicit temporal expressions, such as the names of weekdays (e.g., "on Thursday") or months (e.g., "in July") or references to such points in time like "next week" or "last Friday".

Relative temporal expressions may even include more vague temporal information. Instances of these include phrases such as "in a few weeks" or "some years ago". In general, there is less confidence in determining relative temporal expressions than in explicit or implicit expressions, an aspect that becomes important when all temporal expressions discovered in a document are represented in a temporal document profile, as illustrated in the next section.

Although it might seem almost infeasible to detect and in particular anchor implicit temporal expressions, there have recently been significant advances in detecting and mapping instances of various types of implicit temporal information.

3.6 Temporal Document Profiles

In the following, we describe how explicit, implicit, and relative temporal expressions determined by an information extraction approach are made explicit for our timeline construction. In our time annotated document model, this process of entity extraction is a function denoted *tdp*, for *t*emporal *d*ocument *p*rofile. It associates with each document d of a document collection $\mathcal{D}$ a list of 3-tuples. It has the signature

$$tdp\colon D \rightarrow [E \times C \times P]^*$$

where the set $E = E_e \cup E_i \cup E_r$ denotes the set of explicit, implicit, and relative temporal expressions E_e, E_i, and E_r respectively. C denotes the set of chronons from timelines in $\mathcal{T} = \{T_d, T_w, T_m, T_y\}$. The set P denotes the set of positions of temporal expressions in a document. A position is a composite of the number of the sentence in which the expression occurs, the position in that sentence, and the absolute position of the expression in the document. More precisely, for a particular mapping

$$
\begin{aligned}
d \rightarrow [&(e_1, c_1, p_1), \ldots, (e_{k_e}, c_{k_e}, p_{k_e}),\\
&(e_{(k_e+1)}, c_{(k_e+1)}, p_{(k_e+1)}), \ldots, (e_{k_i}, c_{k_i}, p_{k_i}),\\
&(e_{(k_i+1)}, c_{(k_i+1)}, p_{(k_i+1)}),\\
&(e_{(k_i+2)}, c_{(k_i+2)}, p_{(k_i+2)}), \ldots, (e_{k_r}, c_{k_r}, p_{k_r})]
\end{aligned}
$$

where *tdp* is applied to a document $d \in \mathcal{D}$, resulting in a list of 3-tuples, the meaning of the components of the triples is as follows:

1. The first k_e tuples describe the explicit temporal expressions that have been determined in d, together with their normalized chronons and positions in d (we will elaborate on normalization below).

2. The next k_e+1 to k_i tuples describe implicit temporal expressions, again with their normalized chronons and positions in d.

3. The $k_i + 1$ tuple corresponds to the timestamp $d.ts$ of the document d. It is assumed that every document has such a timestamp. Depending on the confidence one has in the document timestamping approach, the timestamp can be considered either an implicit or explicit temporal expression. For example, if it is known that the document creation times are exact, then the document timestamp should be considered as an explicit temporal expression.

4. Finally, the remaining $k_i + 2$ to k_r tuples describe relative expressions, again together with their normalized chronons and positions.

In the following, for a document $d \in D$, we denote the list of 3-tuples the *temporal document profile* of d, denoted $tdp(d)$.

There are some important properties of a temporal document profile that need to be recognized. First, all chronons $c_i, i = 1 \ldots l$, are *normalized*. That is, all chronons that are elements of the same timeline $T \in \mathcal{T}$ have the same format. For example, all day chronons that have been associated with temporal expressions are represented in the day/month/year format, such as "15/04/1966". Similar formats are assumed for months, years, etc. This normalization step has

some problems of its own like normalizing time zones, which we do not address in this thesis as they are not essential to our approach. As we will see in Chapter 5, respective normalized chronons will serve as labels for our time-based clustering technique.

Second, a chronon $c \in T$ for some $T \in \mathcal{T}$ can be associated with many explicit, implicit, and relative temporal expressions. In fact, the same chronon can even occur several times in a single temporal document profile $tdp(d)$ but then at different positions in the document d.

In summary, a temporal document profile makes explicit all temporal expressions in a document, specified as chronons of well-defined timelines in $\mathcal{T}$. These chronons are then used to construct a timeline for the documents at different levels of time granularity.

Correctly recognizing time information is a key part of our framework. In the next section we provide a summary of temporal information extraction techniques.

3.7 Temporal Information Extraction

Recently there has been a lot of interest in natural language processing applications like text summarization and question-answering that depend on identification of time information. In this section we provide an overview of the most important concepts, techniques, and tools for temporal information extraction. An excellent book that covers the state-of-the-art is Mani *et al.* [66].

The notion of time identification appears as part of the Named Entity (NE) tagging subtask within the Message Understanding Conference (MUC), in particular MUC-6 [38]. In this task, the systems were to recognize (by using SGML tags) entities like named persons, organizations, locations, dates, times, currencies, and percentages. The quality of the extractor is measured in terms of precision and recall according to given standards. In MUC-6 date and time (and day) expressions were labeled using a `TIMEX` tag. In this task only absolute time expressions were required. In the following MUC-7 relative time expressions ("last October") were also part of the task. The next section provides more details on tagging temporal expressions.

A limitation of the date and time NE tasks in MUC-6 and MUC-7 is that there was no way to evaluate those temporal expressions to obtain a precise time. According to the MUC-7 TIMEX tagging guidelines, an expression like "yesterday" in a document about "October 20, 1998" would be tagged as a TIMEX of type `DATE`. An application needs to know that "yesterday" is "October 19, 1998". The TIMEX2 tagging guidelines address this issue by adding a calendrical value for every expression as an attribute of the tagged element.

As discussed in *Mani et al.* [66], it is important to evaluate a relational or indexical time expression and return a calendrical time value. There is utility in separating the evaluation process in mapping the time expression and its evaluation. For example, last Thursday is mapped into the expression *thursday(predecesor(ts(d))*, where *ts(d)* is the metadata that represents the document creation time. In the second phase, a final time is computed based on the value of *ts(d)*. The benefit of this approach is that the semantic interpretation of time is separated from anchoring.

Temporal extraction is not just the identification of times, durations, and frequencies. It also means positioning the events and states with respect to these times and to each other. What to annotate? Temporal annotation involves correctly annotating the temporal position of all temporal entities in a text, in other words, all things that happen or are situated in time.

3.8 TimeML

Because we are dealing with language, it is important to follow annotation guidelines. Examples of such specifications are TIMEX (MUC), TIMEX2, TIMEX3, and TimeML [101, 84]. The earlier work on tagging was performed using SGML tags with no specific DTD in mind.

Recently, TimeML has emerged as the standard markup language for events and temporal expressions in natural language. TimeML has the following interesting properties:

(i) Time stamping of events.

(ii) Ordering events with respect to one another.

(iii) Interpretation of partially determined expressions. Like any markup language, it has a DTD and XML schema.

TimeML has a number of tags described as follows. `EVENT` annotates events in a text. Any event that can be temporally anchored or ordered is captured in the tag. `SIGNAL` is used to annotate temporal functions words like “after”, “during”, etc. There are three subtypes of `LINK`: temporal links (`TLINK`), subordination relationships (`SLINK`), and aspectual connection (`ALINK`). The `TIMEX3` tag is primarily used to mark up temporal expressions, such as times, dates, durations, etc. The standard has four types to denote time, date, duration, and set respectively.

An expression that receives the `TIME` type is one that refers to a time of the day, even if in a very indefinite way. For example: “ten minutes to eleven”, “half past noon”, “6 A.M. Monday, March 12, 2007”, and “nine in the morning”. The `DATE` type can thought of as any extension that

refers to a calendar time. For examples: “Monday, March 12, 2007”, “January 1998”, “yesterday”, or “last week”. To distinguish between TIME and DATE, it is important to look at the granularity of the expression. If the granularity of the expression is smaller than a day, then the expression is of type TIME. An expression is a DURATION if it explicitly describes some extent of time. For example: “24 hours”, “three weeks”, or “12 days in June”. Finally, the SET type is used for expressions that describe a set of regularly reoccurring times, for example, “twice a week” or “every three days”. Below are instances of TIME, DATE, DURATION, and SET.

```
<TIMEX3 tid="t1" type="TIME" value="1998-02-13T14:35:00"
temporalFunction="false"
functionInDocument="CREATION_TIME">02/13/1998 14:35:00</TIMEX3>
<TIMEX3 tid="t2" type="DATE" value="1998-08" temporalFunction="true"
functionInDocument="NONE" anchorTimeID="t41">August</TIMEX3>
<TIMEX3 tid="t3" type="DURATION" value="P1M" temporalFunction="false"
functionInDocument="NONE">a month</TIMEX3>
<TIMEX3 tid="t4" type="DATE" value="PRESENT_REF" temporalFunction="true"
functionInDocument="NONE" anchorTimeID="t41">current</TIMEX3>
<TIMEX3 tid="t5" type="SET" value="XXXX-WXX-1TNI" temporalFunction="true"
functionInDocument="NONE" anchorTimeID="t189" quant="" freq="7D">
Monday</TIMEX3>
```

3.9 Document Annotation Pipeline

We explore the identification of temporal expressions in more detail by providing a document-processing pipeline that includes a number of operations as follows. Given a document collection, the first step is to extract time-related metadata from the documents. This is either the creation or last modified date for a document file or the timestamp provided by the Web server for Web pages or operating system.

The second step is to run a POS tagger on every document. The tagger determines the parts of speech assigned to each word/token like noun, verb etc. in a document.

The tagger also tags sentence delimiters, which are needed later for temporal document annotation. The third step is to run a temporal expression tagger on the POS-tagged version of the document. This step makes the temporal expressions, which are based on the TimeML standard, in a document explicit by producing an XML-like document. That is, resulting documents contain temporal document profile information as XML markup.

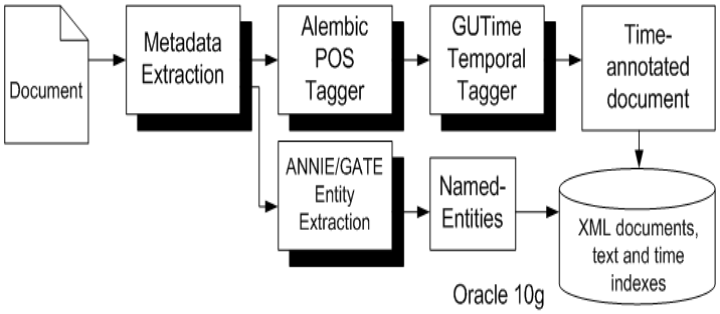


Figure 3.6: Document processing pipeline and index creation.

In a parallel step, a traditional entity extraction component extracts named-entities that later augment the document index. An index is created on the documents that allows to efficiently search for text as well as temporal expressions and other named entities. Figure 3.6 summarizes this document annotation process.

We implement the temporal document annotated pipeline with existing technologies as follows. Corpora processing involves taking a document collection (or hit list from a search application) and processing the documents using the Alembic [3] part of speech tagger (POS tagger). Next, the GUTime [45] temporal tagger recognizes the extents and normalized values of temporal expressions, producing an XML document. Extra entity-extraction is performed by the ANNIE/GATE tool [31] for the identification of other entities.

We use an Oracle $10g$ database for storing the documents that have been annotated using XML. That is, temporal document profiles are embedded within the original documents as XML markup and are not managed outside the documents. The `XMLType` data type in Oracle supports the indexing of document fragments as well as text searching. XML extensions to SQL such as the `extractValue()` and `contains()` functions are used to query annotated documents in the collection using Oracle SQL.

There are a number of POS and temporal taggers available in products and open-source projects. As our emphasis is on using these technologies as components rather than an in-depth study on tagger quality, we made the choice on a very practical aspect: easy to use and format flexibility. This combination allows us to run batch pre-processing sessions on large corpora to get temporal document annotations.

The same goes for named-entity extraction software packages and information retrieval toolkits. The goal here is to show that it is possible to automate the processing of document

collections. A comparison in terms of quality and performance of each component against other alternatives is out of the scope for this work.

Let us take a closer look at each step of the document processing pipeline. We start with an original document represented in plain text like the following:

```
Sunday, 09 July 2006 Italy won the World Cup tonight after beating
France on penalties.
```

As described above, the first step involves using a POS tagger. The Alembic POS tagger returns the original document marked-up as follows:

```
<s>
 <lex pos="NNP">Sunday</lex><lex pos="COMMA">,</lex><lex pos="CD">09</lex>
 <lex pos="NNP">July</lex><lex pos="CD">2006</lex>
 <lex pos="NNP">Italy</lex><lex pos="VBD">won</lex><lex pos="DT">the</lex>
 <lex pos="CD">2006</lex><lex pos="NNP">World</lex><lex pos="NNP">Cup</lex>
 <lex pos="RB">tonight</lex><lex pos="IN">after</lex>
 <lex pos="VBG">beating</lex><lex pos="NNP">France</lex>
 <lex pos="IN">on</lex><lex pos="NNS">penalties</lex><lex pos="PERIOD">.
</s>
```

In the above example, the following tags stand for `NNP` (noun, proper, singular), `CD` (numeral, cardinal), `VBD` (verb, past tense), `RB` (adverb), `VBG` (verb, present participle or gerund), `IN` (preposition or conjunction, subordinating), and `NNS` (plural nouns) respectively.

Note the identification of sentences through `<s> </s>` tags, which are very important for deriving the context of temporal expressions. The second step is to identify temporal expressions. Using the POS version of the document, the GUTime tagger then produces a document marked-up in TimeML that reflects all temporal information tagged appropriately. For the above POS document example, the following document is obtained.

```
<s>
<TIMEX3 tid="t1" TYPE="DATE" VAL="20060709">Sunday, 09 July 2006</TIMEX3>
Italy won the World Cup <TIMEX3 tid="t2" TYPE="DATE" VAL="20060709TNI">
tonight</TIMEX3>after beating France on penalties.
</s>
```

Note that in this marked-up document, the implicit temporal expression “tonight” has been correctly anchored in time; the aspect of deriving such an anchoring for an implicit expression is also indicated by the suffix `TNI` (“night time”) in the `VAL` component and denotes a particular

type of implicit temporal expression. The output of the markup step is the original document with temporal expressions annotated according to the TimeML specification.

For entity extraction, a Java program based on ANNIE produces named-entities such as the identification of locations or countries ("Italy" and "France" in the sports example). This is the last step of the document annotation process. Another example using Einstein's page in Wikipedia can be found in the Appendix A.

In summary, after applying this pipeline to each document of the original collection, the collection is now enriched by encodings of explicit and implicit temporal expressions (and respective time chronons).

3.10 Evaluation

In this section we perform two experiments to asses the quality of the temporal document annotation pipeline. The objective of these experiments is to show that the temporal annotation pipeline provides an effective tool for extracting temporal information from different types of documents.

3.10.1 TimeBank

The goal of our first experiment is to demonstrate that the temporal annotation pipeline we implemented can effectively detect diverse types of temporal expressions. We test this document processing step by using the TimeBank 1.2 corpus. This corpus contains 183 news articles that have been manually annotated using TimeML with temporal expressions related to events, times, and temporal links between events and times [84]. TimeBank is a gold-standard human-annotated corpus and has been used to test quality of temporal extraction.

The task is then to run a plain text version of TimeBank that does not contain any of the original TimeML markup through the annotation pipeline and compare the results against the original TimeBank data set. Running the plain (non-annotated) TimeBank collection through our pipeline resulted in a precision of 87% in terms of temporal expressions that were detected and extracted. Note that we do not intend to evaluate or compare the quality of the temporal tagger. Instead, we focus on the ability to process a document, and to detect and record temporal expressions in temporal document profiles.

This experiment shows that our system can process a collection of documents automatically and effectively detect temporal expressions that are embedded in the documents' content.

In summary, after applying this pipeline to each document of the original document set, the documents are now enriched by encodings of temporal expressions (and respective time chronons). The entire set of documents in TimeML is loaded into a table in the Oracle database. All temporal document measures are computed once the data is in the database. Finally, an XML index and text index are created that allow to efficiently query for document fragments and TimeML content, respectively.

3.10.2 Random Documents

In the second experiment we manually selected twenty random pages from the Web and compared the number of temporal expressions that our technique detected. The process involves reading the content of a document and identifying temporal expressions that the reader (reviewer) thought were important. This is obviously a very intensive and tedious procedure so it does not scale for large collections.

The task requires processing the document through the temporal annotated pipeline and computing its precision. In this context, a manual judgment is used for counting the number of temporal expressions in the original document. The precision is the number of temporal expressions that were automatically detected versus the number that was manually computed. The following is a snippet of the CNN 9/11 news article page (Figure 3.7 shows a screenshot of the original page) that shows some of the temporal expressions found.

```
<TIMEX3 VAL="20010911">9/11/2001</TIMEX3>
<TIMEX3 tid="t1" TYPE="DATE" VAL="20010911">September 11</TIMEX3>:
Chronology of terror
<TIMEX3 tid="t2" TYPE="TIME" VAL="T0845">8:45a.m.</TIMEX3> (all times
are EDT): A hijacked passenger jet, American Airlines Flight 11 out
of Boston, Massachusetts, crashes into the north tower of the World
Trade Center, tearing  a gaping hole in the building and setting it
afire.
<TIMEX3 tid="t3" TYPE="TIME" VAL="T0903">9:03 a.m.</TIMEX3>: A second
hijacked airliner, United Airlines Flight 175 from Boston, crashes
into the south tower of the World Trade Center and explodes. Both
buildings are burning.
```

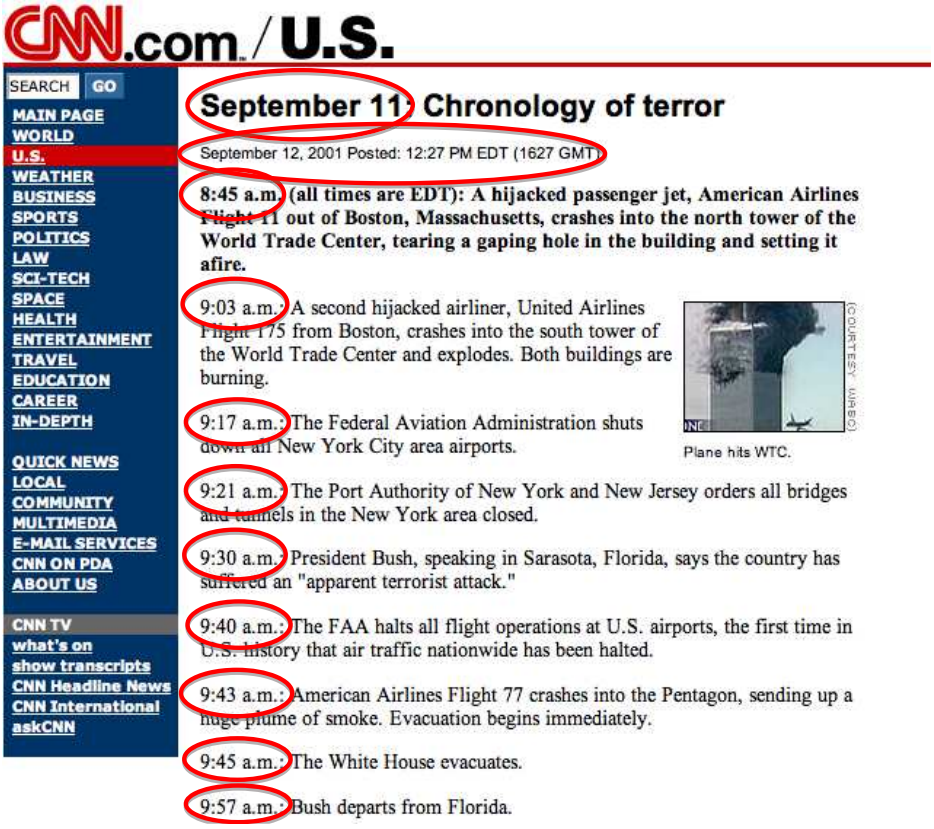


Figure 3.7: CNN news article for September 11, 2001. From CNN.com.

Table 3.1 shows the metrics for Einstein's page in Wikipedia, the CNN 9/11 news article, and a hypothetical résumé from a job website. Once again, the goal of this experiment was to test how viable our approach is using some random documents with no pre-existing judgments on their temporal content.

It should be noted that our prototype is capable of handling arbitrary documents and collections efficiently. In particular, it also allows to extend the rule base for the temporal tagger in order to accommodate previously unknown temporal expressions that are to be mapped to chronons, such as diverse types of holidays or events mentioned in documents. In this sense, our prototype system provides a comprehensive infrastructure for the processing of documents and collections that are subject to temporal analysis and exploration.

Object	Temporal expressions	Temporal expressions detected	Precision
Albert Einstein page in Wikipedia	338	303	0.896%
CNN 9/11 news article	78	76	0.974%
Résumé from a job site	23	19	0.82%

Table 3.1: Precision for three documents.

3.11 Related Work

There is exciting research on adding a time dimension to certain applications like news summaries [4], temporal patterns [93], and temporal Web search [81]. The special issue on temporal information processing gives a clear map of current directions [67]. Using temporal information for query formulation and evaluation is presented in Diaz and Jones [34]. The goal is to extract metadata from documents and use it as part of temporal profiles for queries, which is important for query precision analysis. There is also a proposal for a time ontology for the Semantic Web [51]. Retrieval of time information has been an area of work on the medical informatics community [50].

Research in the information extraction area ranges from natural language processing where the main objective is to measure the quality of the extractor to a more practical approach where IE is just a function of a bigger document processing program. For example, the SphereSearch search engine uses entity extraction as a mechanism for automatically annotating documents [43]. Another example is using annotations for augmenting XML documents and improving fragment retrieval is Chu-Carroll *et al.*[28]. When dealing with time, a key component of any IE system is the ability to extract and normalize temporal data, , which is presented by Childs *et al.* in [27].

Research on temporal annotations is very recent, and it is well covered in the book by Mani *et al.* [66]. Identification of time depends heavily on the language and the corpora, so traditional information extraction systems tend to fall short in terms temporal extraction. Based on the latest advances, new research is emerging for automatic assignment of document event-time periods and automatic tagging of news messages using entity extraction described by Schilder and Habel [91] and Schilder [92]. An example of guidelines for annotation temporal information is Mani *et al.* [68]. More sophisticated annotation schemes are presented by Muller and Tannier in [77]. An example of tools for annotation is TARSQI by Verhagen *et al.* in [109].

Temporal databases are databases that store information about states of the real world

across time. Such databases include a temporal data model and extensions to a structured query language. A survey of time in databases is presented by Ozsoyoglu and Snodgrass in [80].

3.12 Summary

We have introduced the foundations of our temporal information retrieval approach. As a first step in that direction, we presented a detailed discussion about of time, its role in information retrieval, and how temporal information can be found in documents.

We presented the concepts and techniques underlying a novel document exploration framework in which we utilize temporal information extracted from documents.

The proposed approach relies on extracting diverse types of temporal information from documents and making this information explicit in the form of temporal document profiles. Such profiles, obtained in a document annotation process, are used to construct timelines at different levels of time granularity. These timelines build the basis for anchoring documents, and provide a means for analyzing and exploring temporal features of document collections, such as browsing or hit list based search results.

We designed a system for annotating documents with temporal information. The implementation shows that the proposed technique can effectively be realized using existing tools and systems. We conducted an evaluation of the feasibility of our method that showed very good precision. We also outlined the main concepts of temporal information extraction and the TimeML standard, which are important parts of the temporal document annotation pipeline.

Chapter 4

Time-based Document Analysis

In the previous chapter we studied how time is represented in documents and defined a model for annotating document collections with temporal information. We implemented a prototype that showed how to automatically process different types of documents. Now that the necessary underlying infrastructure is in place, we can take a look at a set of documents from a temporal perspective.

In this chapter we study models and techniques to analyze a document based on its temporal content. The goal of this approach is to provide a temporal characterization of a document that can serve as the basis for comparison and exploration of a collection of documents.

This chapter is organized as follows. The next section motivates our approach and details the objectives. Section 4.2 presents novel temporal document measures such as temporal richness and specificity, among others. Section 4.3 introduces a temporal distance function. Section 4.4 details where temporal expressions appear at the sentence level. Section 4.5 presents examples that illustrate the analysis process of documents. Finally, Section 4.6 discusses related work.

4.1 Motivation and Objectives

In the previous chapter we showed how temporal information can be expressed in the English language in different ways. We proposed the concept of a temporal document profile to explicitly and concisely represent time related information in a document. We also demonstrated how to annotate documents with time information.

With all that information at hand, we would like to further analyze a document collection

to discover patterns. How can we make sense of the temporal features of documents? How can we exploit these essential features for document search and exploration purposes?

Temporal information embedded in a document might tell what type of document it is, such as news article, biography, résumé, a game report, etc. Let us examine a few sample documents to get a better idea of how time related expressions appear in different types of documents.

We start with the CNN 9/11 news article that describes the September 11, 2001 attacks (Figure 3.7). The text concentrates on the timeline of events for that particular day so the type of temporal information is mainly time and date, with a high percentage of time to the minute precision. We want to have this distinction between the granule time and date, as we discuss later in Section 4.2.

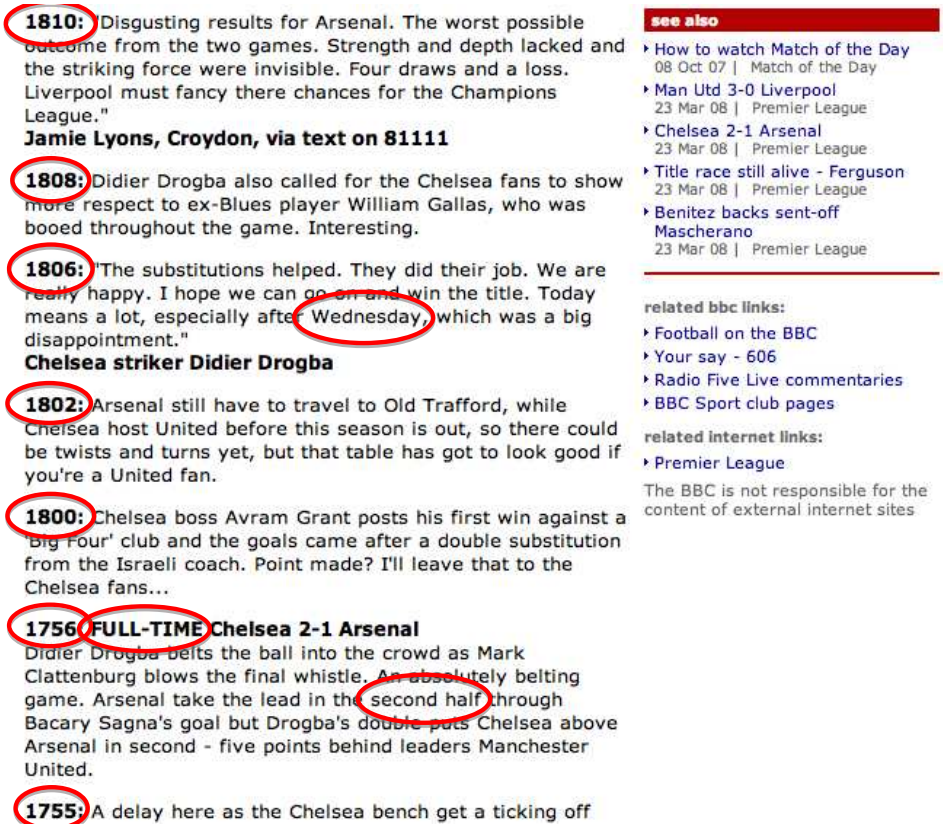
The structure of the document is typical of a news article, which contains a descriptive title, timestamp, and the text. In this particular example, time appears in the title as date and in every paragraph as time, which is a granule finer than date. As the title explains it, `September 11: Chronology of terror`, the content is about a detailed chronology and the events are presented in sequential order as shown in the following excerpt from the Web page.

```
September 11: Chronology of terror
September 12, 2001 Posted: 12:27 PM EDT (1627 GMT)

8:45 a.m. (all times are EDT): A hijacked passenger jet, American Airlines
Flight 11 out of Boston, Massachusetts, crashes into the north tower of
the World Trade Center, tearing a gaping hole in the building and
setting it afire.

9:03 a.m.: A second hijacked airliner, United Airlines Flight 175 from
Boston,crashes into the south tower of the World Trade Center and explodes.
Both buildings are burning.
```

A different category of news is sports, where there is quite a bit of time information. In a game, there are a number of important events along with duration, and their time-based description at a fine level of granularity, and their timely description is crucial to better understand the final score. There are usually two types of game descriptions: a *recap* that highlights the main events and a *step-by-step* that covers play by play. Interestingly, this is common for a lot of sports.

As examples, we show a step-by-step of the last few minutes of a football (soccer) game (Figure 4.1) and a recap of a basketball game (Figure 4.2). As expected, the step-by-step document

has detailed minute description (the basketball equivalent is even more precise) including some specific temporal expressions that are relevant to the game (e.g., full-time, added time). The recap is closer to a typical news article and highlights the main events. Again, certain temporal expressions are only valid in this context like "fourth quarter".



1810: Disgusting results for Arsenal. The worst possible outcome from the two games. Strength and depth lacked and the striking force were invisible. Four draws and a loss. Liverpool must fancy there chances for the Champions League."
Jamie Lyons, Croydon, via text on 81111

1808: Didier Drogba also called for the Chelsea fans to show more respect to ex-Blues player William Gallas, who was booed throughout the game. Interesting.

1806: "The substitutions helped. They did their job. We are really happy. I hope we can go on and win the title. Today means a lot, especially after Wednesday, which was a big disappointment."
Chelsea striker Didier Drogba

1802: Arsenal still have to travel to Old Trafford, while Chelsea host United before this season is out, so there could be twists and turns yet, but that table has got to look good if you're a United fan.

1800: Chelsea boss Avram Grant posts his first win against a 'Big Four' club and the goals came after a double substitution from the Israeli coach. Point made? I'll leave that to the Chelsea fans...

1756: FULL-TIME Chelsea 2-1 Arsenal
Didier Drogba belts the ball into the crowd as Mark Clattenburg blows the final whistle. An absolutely belting game. Arsenal take the lead in the second half through Bacary Sagna's goal but Drogba's double puts Chelsea above Arsenal in second - five points behind leaders Manchester United.

1755: A delay here as the Chelsea bench get a ticking off

see also
- How to watch Match of the Day
  08 Oct 07 | Match of the Day
- Man Utd 3-0 Liverpool
  23 Mar 08 | Premier League
- Chelsea 2-1 Arsenal
  23 Mar 08 | Premier League
- Title race still alive - Ferguson
  23 Mar 08 | Premier League
- Benitez backs sent-off Mascherano
  23 Mar 08 | Premier League

related bbc links:
- Football on the BBC
- Your say - 606
- Radio Five Live commentaries
- BBC Sport club pages

related internet links:
- Premier League

The BBC is not responsible for the content of external internet sites

Figure 4.1: Football (soccer) game transcript. Adapted from BBC Sports.

A very different type of document is a résumé or curriculum vitae that describes a professional career by milestones and time periods (Figure 4.3). The professional progression is usually covered by time spans and presented in a chronological order starting from the current position. Education or any other major achievement are described by the year they were accomplished. A similar type of document is an author's entry on the DBLP bibliography data set, which lists publications by year [33]. DBLP (Digital Bibliography & Library Project) is a computer science bibliography Website that indexes more than one million articles and contains more than 10,000 links to home pages of computer scientists.

Spurs Complete Sweep to Capture Fourth NBA Title
By TOM WITHERS, AP
Sports Writer Posted Jun 15 2007 12:34AM

CLEVELAND, June 14 (AP) -- True roundball royalty, the San Antonio Spurs are once again wearing the crown.

MVP Tony Parker scored 24 points, Manu Ginobili had 27 - 13 in the fourth quarter - and the Spurs moved in among the NBA's greatest franchises with an 83-82 victory Thursday night for a sweep of the Cavaliers.

With their fourth championship since 1999 - and third in five years - the Spurs joined the Boston Celtics, Los Angeles Lakers and Chicago Bulls as the only teams in NBA history to win four titles.

Figure 4.2: NBA 2007 Finals. Excerpt from NBA.com

The examples presented so far are illustrations of how time information in documents can provide some insight into a document's type. It would be helpful to have a useful technique to find an analytic way or systematic approach that can produce a temporal characterization of a document.

A first step is to study temporal properties of documents by performing a detailed analysis of the occurrences of temporal expressions in a document. To summarize, the objectives of this chapter are to:

(i) Define the concept of temporal document similarity.

(ii) Provide a number of temporal aggregate measures such as richness and specificity, among others.

(iii) Investigate the distribution of temporal expressions as they occur sequentially in a document.

(iv) Explore query scenarios that can benefit from temporal ranking.

(v) Provide a number of case studies that exemplify the main concepts introduced.

(vi) Develop an evaluation framework and present experimental results.

```
John Doe

Work Experience

Bay Area Hospital, CA 10/91-Present
Director, Clinical Services (5/91-Present)
Director, Emergency Psychiatric Services and Day Programs (8/88-5/91)
Psychologist (6/82-6/86)

Education

University of California, Master's Degree in Counseling, Psychology, 1982
University of California, Bachelor's Degree in Psychology, 1980
```

Figure 4.3: Hypothetical résumé.

4.2 Temporal Measures in Documents

If we take yet another look at documents, from a reader's perspective, we can see that they are made to carry and offer very different stories in very different ways. All these documents cover a specific time period or have some predominant temporal expressions. How can we identify the most noticeable temporal information in a document?

Temporal document profiles provide the basis for deriving measures that better describe documents in terms of their temporal information content. In the Section 4.2.1, we first define some simple aggregate measures for documents such as temporal richness and specificity and then detail more complex measures such as temporal coverage and temporal order in Section 4.2.2. Temporal coverage in particular leads to a novel approach for computing temporal document distances (Section 4.3) and also provide the basis for interesting visualization metaphors, which are discussed in Chapter 6 in the context of document exploration and similarity search applications.

4.2.1 Temporal Aggregate Measures

In the following, we assume a single document $d \in \mathcal{D}$ with temporal document profile $tdp(d)$. Let t-seq$(d) = \langle (c_1, p_1), (c_2, p_2), \cdots, (c_k, p_k) \rangle$ denote the chronon/position pairs of the profile.

The first measure simply defines the ratio of chronons in the document d to the total number of chronons in the collection D. We call this measure *temporal richness* of d, denoted t-rich(d). The higher this ratio, the richer a document d is in its temporal information content. Note

that in the extreme case, a document may only have a document timestamp and no other temporal expressions in its content.

The next measure focuses more on the types of chronons in t-seq(d), in particular the frequencies of types. We extended the original definition of $\mathcal{T}$ as defined in Section 3.4 by adding T_t for time. A document d is said to be *temporally most specific* with respect to chronons of type $T \in \mathcal{T}, \mathcal{T} = \{T_t, T_d, T_w, T_m, T_y\}$, if most of the chronons in t-seq(d) are of type T. For example, a document detailing the events of a football game might be very specific, using chronons of type time (T_t) whereas a résumé document might be more specific with respect to chronons of type year or month (T_y or T_m). In the following, we denote the *temporal specificity* of a document d as t-spec(d). As this measure is simply based on the frequencies of chronons of different types in t-seq(d), it naturally defines an order among the types in $\mathcal{T}$ for a document d, with the first type in that order being the most frequent chronon type in t-seq(d) and so on.

The last aggregate measure considered here describes the *temporal boundaries* of a document. For this, the earliest and the most recent chronons in t-seq(d) are determined. The problem here is that chronons can be of different types and thus the precedence relationship $\prec_T$ might not always be applicable. We therefore make the following assumption: If two chronons $t_i \in T_i$ and $t_j \in T_j$ from t-seq(d) can each be considered the earliest (most recent) chronon, then the one with the coarser granularity is chosen as the earliest (most recent). Assume, for example, the two chronons "1999" and "March 1999" in t-seq(d) and there is no chronon describing a point in time earlier (more recent) than 1999. In this case, we choose the chronon "1999" as the earliest (most recent) one. In the following, we denote the earliest and most recent chronons in t-seq(d) as t-low(d) and t-high(d), respectively.

For example, Einstein's Wikipedia page is most specific with respect to year chronons, followed by day chronons. In terms of the other metrics, we have t-low = "1879/14/3" (his date of birth), and t-high = "2008" (recent events and references related to Einstein).

In the CNN 9/11 news article example, the page is most specific with respect to a time chronon. Because the document is focused on a single event, the values for t-low = "9/11/2008, 8:45 AM" and t-high = "9/11/2008, 11:54 p.m." are very close.

(a) t-seq(d) = $\langle (1, 2007/12/06), (25, 1947), (27, 1879/03/14), (35, 1955/04/18),$
$(124, 1921), (127, 1925), (131, 1929), (161, 1879/03/14), (165, 1955/04/18),$
$(425, 1880), (475, 1893), \ldots, (1271, 1905), (1348, 1906), (1359, 1908),$
$(1365, 1910), \ldots, (7648, 1940), (7651, 1940)\rangle$

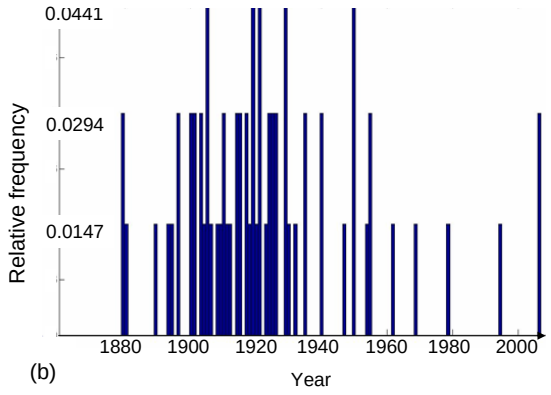


Figure 4.4: (a) Parts of the sequence of position/chronon pairs (t-seq) derived from Einstein's Wikipedia Web page. (b) Temporal coverage of this sequence represented in the form of a histogram.

4.2.2 Temporal Coverage and Order

The above measures are coarse-grained, i.e., they aim at describing *aggregated* information about the temporal content of a document. In the following, we introduce two concepts to better represent the temporal content and its structure at a more fine-grained level. This is accomplished by using (1) a histogram approach to describe the temporal coverage and specificity of a document's content with respect to time periods, and (2) temporal order to represent the occurrences of chronons in document order.

Coverage description using histograms

We use an equiwidth histogram approach to describe the distribution of chronons in a document over a period of time, called *temporal coverage*. The objective is to represent (a) what periods in time are covered by the document's content and (b) what periods in time are emphasized in terms of chronon frequencies. Loosely speaking, for a given document d, the period of time (range) described by its histogram is determined by the temporal boundaries t-low(d) and t-high(d), and the number of values in a bucket is based on the number of chronons that fall in that bucket's subrange (period in time). In the following, we detail the construction of a histogram for a document d and

then describe zooming operations on the histogram in support of document exploration operations.

First, a range for the histogram's buckets needs to be determined. Given that the chronons in t-seq(d) can be of different types, we first need to establish a common domain for the buckets' range. For this, we choose as temporal unit the type of the chronon in t-seq(d) with the coarsest granularity, even though the temporal specificity of the document might be of a finer granularity. Consider, for example, a document d with t-low(d) = "1950" and t-high(d) = "2003". The histogram for d, denoted t-hist(d) then is made up of 54 buckets, each labeled by a year chronon in $[1950, 2003]$. In the worst case, there might only be one bucket in t-hist(d), namely when all chronons refer to the same year (e.g., at the day level) but there is at least one chronon $c \in$ t-seq(d) of type year.

Assume a histogram with ℓ buckets $b_1, b_2, \cdots, b_l$ has been created and the range is based on chronon type $T \in \mathcal{T}$. In the above example, ℓ would be 54 and T = year, that is, buckets are labeled by years. Now, chronons in t-seq(d) are placed into the buckets. A chronon $c_i \in$ t-seq(d) is placed into a bucket b_j if c_i is covered by or is equal to the time period described by the bucket's label.

With each bucket in a histogram two important measures are associated. First, for each bucket $b_i, i \in [1, \ell]$, the relative frequency of the chronons placed in b_i, denoted t-freq(b_i), is maintained. The relative frequency for b_i is the ratio between the number of chronons in b_i to the total number of chronons in all the buckets for the histogram. The domain of the chronons' relative frequency is the interval $[0, 1]$, and the histogram is an approximation of the probability density function representing the underlying distribution of the chronons. Second, as for the whole document d, also with each bucket b_i a temporal specificity, denoted t-spec(b_i), can be determined. For example, it might be that 20 chronons from t-seq(d) have been placed in a bucket with year "1950" but most of these chronons are actually of type day. In Chapter 6, we will show how this information is presented to the user in a graphical user interface for document exploration purposes.

Figure 4.4 illustrates an example of a position/chronon sequence (a) for Einstein's Wikipedia Web page and the temporal coverage (b) derived from that sequence. Note the temporal "focus" of the document for the time period between 1900 and 1930, something that, if appropriately represented to the user, itself already provides some important information about the temporal content of the document.

Note that although the label of a bucket b_i is based on a coarse-grained type T, the bucket might contain several chronons that are of a granularity T' finer than T, i.e., $T' \ll T$. In order to obtain more fine-grained temporal information from such buckets, a *temporal zoom operation* can

be applied as follows. Assume a bucket b_i labeled with a year chronon y. A zoom-in operation partitions this bucket into twelve buckets, each labeled with a month of the year y. The chronons from b_i are then distributed over these twelve new buckets, based on whether the new bucket's label covers that chronon. With each new bucket, again a frequency and temporal specificity is associated. As b_i might contain chronons of granularity T, not all chronons can be distributed over the new buckets. This zoom-in operation can be applied in an iterative fashion, exploring individual buckets at an increasingly fine level of time granularity. We illustrate this important functionality as part of the document exploration process in the context of a graphical user interface in Chapter 6.

Temporal Order

A histogram based temporal coverage naturally describes a document's content using periods in time. It details how such periods are emphasized in terms of chronon distributions, frequencies and temporal specificity. In addition to this *content-specific* measure, we introduce another measure that helps describing how the temporal content and underlying chronons are *ordered* in a document. This measure is called *t-order* and is based on the positions chronons occur in a document.

Assume again t-seq(d) $= \langle (c_1, p_1), (c_2, p_2), \cdots, (c_k, p_k) \rangle$ is given for a document d with document length $len(d)$ and temporal boundaries t-low(d) and t-high(d). A chronological order for d, denoted t-order(d), is constructed in a position/time coordinate system as follows: the range of the x-axis is 1 to $len(d)$, specifying the different token positions in d in increasing order. The range of the y-axis is determined by t-low(d) (as minimum value) and t-high(d) (as maximum value). Conceptually, the units underlying the y-axis are based on the finest granularity of any chronon in t-seq(d). Assume the units underlying the y-axis are based on months (for other granularities the approach is analogous), that is, all chronons in t-seq(d) are of type month or coarser. Each chronon/position pair (c_i, p_i) in t-seq(d) is now processed as follows. If c_i is of type month, then simply a point at p_i, c_i is plotted. If the type of c_i is of granularity coarser than month, then a vertical bar is plotted at position p_i, where the bar covers all the y-values that are contained by c_i. For example, if c_i is of type year, then the bar at the position p_i spans twelve months. Figure 4.5 indicates the temporal coverage and t-order for Einstein page in Wikipedia, in (b) and (c), respectively. Note that in part (b) of the figure there is no bucket with more than 3 chronons.

The rationale for having t-orders in addition to histograms is to describe how time periods represented by a histogram are in fact organized in the document. For example, two documents

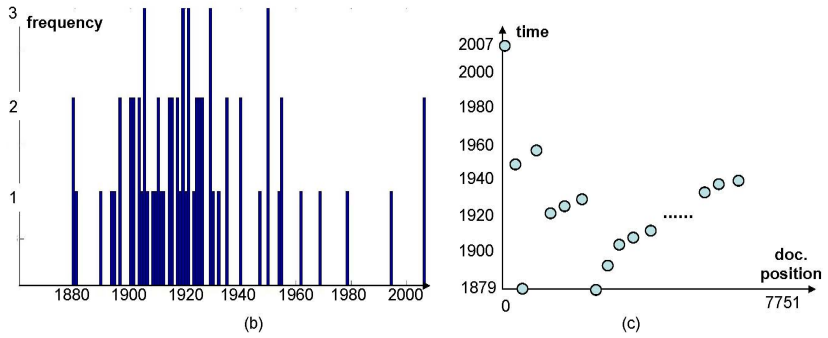


Figure 4.5: Temporal coverage and t-order for Einstein page in Wikipedia.

might have very similar histograms, but their t-orders might look completely different. One t-order, for example, might clearly show that the time period(s) are covered in a true chronological fashion, starting with the earliest chronon at the beginning of the document and having the most recent chronon occurring at the end of the document. In the second t-order, on the other hand, there might be no such patterns of (partial) chronological order. There are certain types of document for which one would expect that many of the chronons derived from the document's content follow such a monotonicity property. Visualizing a t-order in addition to a temporal coverage for a document, of course, might give the user further cues on the temporal information content.

4.3 Temporal Document Similarity

An important feature of our proposed temporal document analysis and exploration framework is to allow users to search for "temporally similar" documents in a hit list or document collection. But what does it mean that two documents are temporally similar? While this question itself warrants an extensive study that goes far beyond the scope of this thesis, we propose a simple, yet intuitive (especially for the user) and easy way to compute the *temporal distance* between two documents. In particular, we develop this measure with a collection of topic specific documents or documents in a hit list in mind. The reason for this is that two arbitrary documents can exhibit similar temporal coverages (e.g., a document about the history of the World's Fair and a document about natural disasters) but from a topic point of view, these are likely unrelated. In a topic specific document collection or a hit list, such a scenario of unrelated documents is less likely to occur.

In the following, we assume a hit list L of documents as result to a user query (the approach works analogously for any topic specific document collection $\mathcal{D}$). Assume the user considers a document $d \in L$ as interesting, because d covers a particular period in time in very detail (this

observation alone can be made from the visual representation of the document's histogram as we show in Chapter 6). If the user is now interested in documents from L that are temporally similar to d, he/she can either look at each document's histogram or simply invoke a distance function that is realized as follows.

The temporal distance is defined as the difference between the distributions of the chronons of two documents. Since we use histogram as the estimation of the underlying distribution of the chronons, we employ a distance measure for ordinal type histograms described in [24]. We modify this approach to take the characteristics of our type of histograms into account. Assume two documents d and d' with histograms t-hist(d) = $b_0, b_1, \ldots, b_d$ and t-hist(d') = $a_0, a_1, \ldots, a_{d'}$, respectively. In order to apply the technique, the two histograms first need to be normalized in terms of underlying chronon type and temporal boundaries. For this, the histogram based on the coarser type of chronon is chosen and the buckets of the other histogram are aggregated correspondingly. For example, t-hist(d') might be based on month chronons while t-hist(d) is based on year chronons. The buckets in t-hist(d') are then aggregated and new buckets, now labeled with year chronons are created. Next, as the temporal boundaries of the two histograms may differ, they need to be brought to the same length. This is easily achieved by extending the histograms to the left and right, as necessary, with empty buckets so that both histograms have the same t-low and t-high values (see Section 4.2.1). Assume this normalization results in the two modified histograms t-hist(d) = $b_0, b_1, \ldots, b_n$ and t-hist(d') = $a_0, a_1, \ldots, a_n$ that now have the same lengths n, temporal boundaries, and underlying chronon type.

After these two histogram normalization steps, the distance function

$$dist(d, d') = \sum_{i=0}^{n-1} \left| \sum_{j=0}^{i} \text{t-freq}_n(b_j) - \text{t-freq}_n(a_j) \right| \tag{4.1}$$

derived from [24] is applied to the two documents d and d' and associated (modified) histograms. This non-metric distance function basically determines the number of re-arrangements of buckets in one histogram so that the two histograms are trivially identical (recall that because of the normalization, both histograms have the same "number" of elements). The closer the computed value to 0, the *temporally similar* the documents are.

It should be noted that standard distance functions such as cosine-based similarity are not applicable here (e.g., when histograms are represented as vectors) as they do not consider the shape but only focus on frequencies. Given a document $d \in L$, the distance to all other documents in L

is computed and, based on the computed values, the remaining documents in L can in fact also be sorted, presenting the user the temporally most similar documents first.

In Chapter 6 we illustrate how temporal similarity can be surfaced as a feature of an exploratory search application. Also in that chapter, we will show experimentation results using a case scenario.

4.4 Sentences Analysis

So far in our analysis, we have looked at a document as a single unit. A document is composed of different structures like sections and paragraphs, which, for example, contain one or more sentences. In linguistics, a sentence is a unit of language, characterized by the presence of a verb. Tense is a specific mechanism built into language for locating information in time.

To identify properly all temporal expressions in a document, it is very important to detect sentences correctly. Once the extraction has been completed, one can look at individual sentences and ask some questions. Where do most of the temporal expressions occur (beginning, middle, end)? Are there sentences with more than one temporal expressions? Is there a particular section of a document where most of sentences that contain temporal expressions appear?

Extracting sentences with temporal information from a document is relatively straightforward. For each sentence we need to count all tokens regardless if they are candidates for a stoplist and compute the position of the temporal expression. The output is then the ratio of position/sentence length for one or more temporal expressions per sentence. For example, `In 1912, Einstein returned to Switzerland to accept a professorship at ETH`, has a ratio of 0.16 (2/12). This ratio indicates that the temporal expression occurs early in the sentence. By also counting the number of sentences (with and without temporal expression) we have a good indication at which part of the document most of the temporal expressions are located.

We use the following computation process to analyze a document:

(i) Read text annotated in TimeML.

(ii) Detect sentences according to the algorithm proposed in Manning and Schütze [70].

(iii) Filter out sentences that do not have `TIMEX3` tags.

(iv) For each sentence with a `TIMEX3` tag: count number of tokens and compute position of temporal expressions. There can be more than one `TIMEX3` tag in a sentence. Detecting the correct

number of temporal expressions in a sentence has implications in the snippet generation that we discuss in the next chapter.

As examples to illustrate the analysis we use Einstein's Wikipedia page and the CNN 9/11 news article. Figure 4.6 shows where temporal expressions occur in both documents. The x-axis represents the sentence number, which is sequential but could be interrupted (not all sentences have temporal expressions). The y-axis shows the ratio of a temporal expression over sentence length. We can see a zigzag effect as we follow the document from start to end.

In the case of the CNN 9/11 news article document, most of the temporal expressions occur early in the sentence, for example, e.g.,

```
9:17 a.m.: The Federal Aviation Administration shuts down ...
9:21 a.m.: The Port Authority of New York ...
```

As we move forward in a document, there are few sentences that include temporal expressions closer to the end (`Giuliani urges New Yorkers to stay home Wednesday if they can`).

In the case of Einstein's page, the first few sentences in the document seems to behave similarly to the previous example (`In 1880, the family moved to Munich, where his father ...`) but there is a high number of sentences that contain temporal expressions at the end. Those sentences are located in the second half of the document and most of them are reference material:

```
A Brief Biography of Albert Einstein, April 2005.
Einstein's Big Idea, Public Broadcasting Service, 2005.
```

In both cases, most of the temporal expressions occur *early* in the sentence and *early* in the document. In the CNN 9/11 news article, the chronons are explicit time. For Einstein's page, there is a high frequency of sentences with the pattern "In *temporal expression* ..." or "During *temporal expression* ...". There are cases where one sentence contains more than one temporal expression (`In 1922 Einstein was awarded the 1921 Nobel Prize in Physics`).

Figure 4.7 shows the distribution of temporal expressions within a sentence. In the CNN 9/11 news article, most of the temporal expressions occur in the first 10% of a sentence. In the case for Einstein's Wikipedia page, there is a significant portion in the

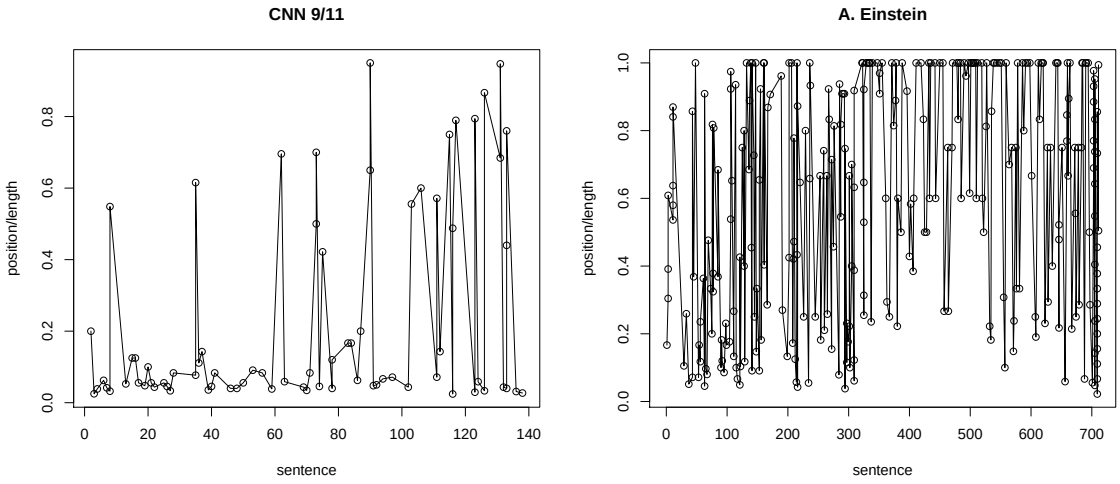


Figure 4.6: Temporal expressions as they appear in sentences.

30% mark but a big spike at the end. Once again, a large number of references (around 99) with the format "Publication, Year" dominates the last part of the document.

4.5 Case Studies

In this section, we present the experiments we conducted using the techniques described in the previous sections and discuss the results of individual experiments. The objective of these experiments is to show that (1) different document collections vary greatly in terms of temporal richness and thus temporally poor collections are not suitable for temporal document exploration, and (2) that our temporal document metrics and document similarity provide a basis for useful temporal document analysis and exploration tasks.

4.5.1 TimeBank

The objective of the experiment with TimeBank was to get an idea of the temporal measures in a manually temporal annotated corpus of 183 documents. TimeBank was introduced earlier in subsection 3.10.1 as an example of TimeML data set available for research [101].

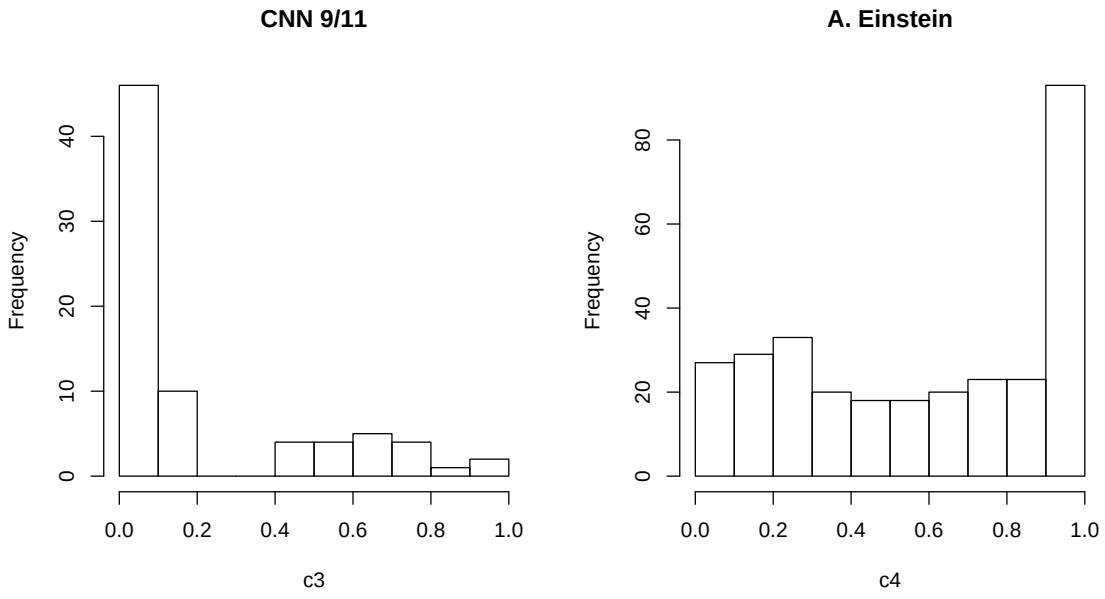


Figure 4.7: Frequency of ratio position/sentence length for two documents.

The temporal analysis of this collection shows that the temporal specificity is restricted to date, which is expected as the collection only contains news documents from prominent sources. Another interesting point is that the bulk of the documents cover the years 1988 to 1998, as illustrated in Figure 4.9. This is also consistent with the description of the collection. Table 4.1 shows a summary of the results and Figure 4.8 shows the distribution of temporal expression in the TimeBank document collection. TimeBank has 62,009 words (tokens) and 1,414 temporal expressions. One can say that temporal expressions account for 2% of the collection or that there is one temporal expression per 50 words (1/50).

4.5.2 Featured Articles in Wikipedia

The goal of the next set of experiments is to perform a temporal analysis of a collection that includes a diverse set of documents in terms of their content. The previous analysis using news articles showed that the collection is not that rich, temporally speaking. But how does such an analysis look like for a different collection, e.g., historical documents?

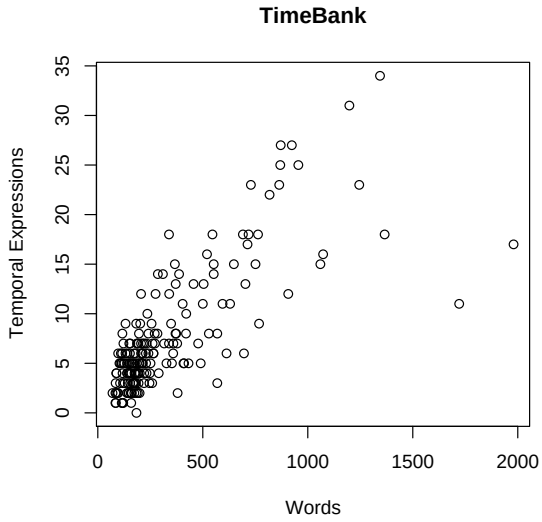


Figure 4.8: Distribution of chronons in TimeBank collection; x-axis: length of document in words, y-axis: number of temporal expressions.

Temporal measure	**Value**
Min number of chronons per document	1
Max number of chronons per document	34
Mean number of chronons per document	7.7072
Number of time type chronons	44
Number of day type chronons	559
Number of month type chronons	106
Number of year type chronons	177
Number of other type chronons	503
Temporal specificity	day
t-low	1901
t-high	2011/03/01

Table 4.1: Temporal measures computed for the TimeBank 1.2 document collection.

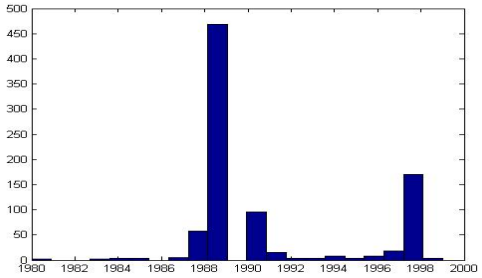


Figure 4.9: TimeBank histogram from 1980 to 2000; x-axis year, y-axis: temporal expressions.

How do analysis results vary among different types of (topic specific) collections? For example, what is the difference between a collection of sports documents compared to a collection of geography documents?

Wikipedia is a multilingual, Web-based, free content encyclopedia project, operated by the Wikimedia Foundation, a non-profit organization [114]. Wikipedia is one of the most popular destinations on the Web for free content maintained by volunteers around the world. Based on Wiki technology, its search features are very limited and they do not take advantage of all the rich content available. At time of writing, Wikipedia has approximately 7.8 million articles in 253 languages, 1.8 million of which are in the English edition.

Wikipedia is a good candidate for such a comprehensive analysis but, at the same time, the quality of articles varies. A high quality subset of articles with excellent content are "featured articles". Featured articles are considered to be the best articles in Wikipedia, as determined by Wikipedia's editors [14] who review them according to accuracy, neutrality, completeness, and style.

At the time of this particular experiment there are 2,050 articles organized in 28 categories (shown in Table 4.2, with number of documents (size), maximum, minimum, and total number of temporal expressions). These categories cover a wide range of topics so it makes this collection a high quality subset of Wikipedia.

Using the existing infrastructure introduced in Section 3.9, we downloaded the set of featured articles and determined, based on the extracted temporal document profiles, how

Id	Category Name	Size	Total	Min	Max
1	Art, architecture, and archaeology	70	7654	34	272
2	Awards, decorations, and vexillology	26	2252	20	206
3	Biology and medicine	170	19191	21	347
4	Business, economics, and finance	19	2726	38	338
5	Chemistry and mineralogy	30	3642	21	331
6	Computing	17	2698	15	325
7	Culture and society	45	6621	34	313
8	Education	34	7348	57	433
9	Engineering and technology	35	3546	13	255
10	Food and drink	11	1078	55	234
11	Geography and places	154	31549	27	544
12	Geology, geophysics, and meteorology	78	8354	23	440
13	History	147	19555	16	415
14	Language and linguistics	17	1708	16	265
15	Law	28	2977	29	363
16	Literature and theatre	121	17707	14	417
17	Mathematics	13	1234	21	215
18	Media	168	25318	22	550
19	Music	166	25600	38	659
20	Philosophy and psychology	13	1638	11	271
21	Physics and astronomy	73	10009	16	321
22	Politics and government	62	11132	29	717
23	Religion, mysticism, and mythology	38	4924	39	439
24	Royalty, nobility and heraldry	87	8736	27	305
25	Sports and recreation	136	26130	44	664
26	Transport	52	6924	32	454
27	Video gaming	84	10953	27	421
28	Warfare	156	18547	12	361

Table 4.2: Categorization of featured articles. Size represents the number of documents in a category. Total, min, and max are computed over number of temporal expressions per category.

temporally rich the collection is. This is important because in certain cases, as indicated above, a temporal analysis and subsequent exploration might not be very useful if there is a low number of temporal expressions in the collection.

Figure 4.10 shows the distribution of temporal expressions in featured articles, after processing the document collection. The x-axis represents the size of the document in tokens, and the y-axis the number of temporal expressions. Compared to the TimeBank data set, this experiment clearly shows that featured articles are much richer in terms of temporal expressions.

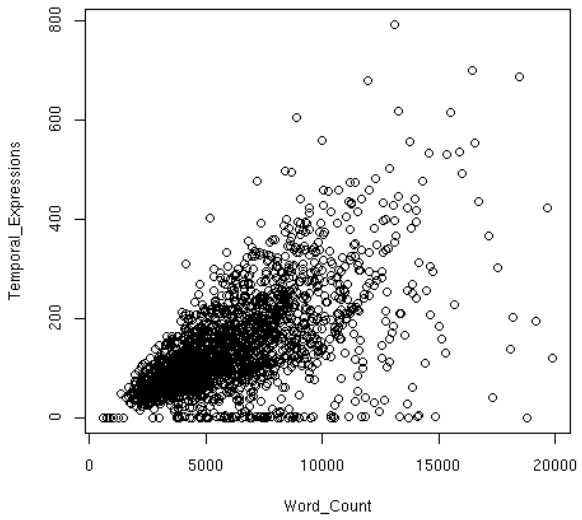


Figure 4.10: Distribution of temporal expressions in the 2,050 featured articles in Wikipedia.

A breakdown of the distribution of temporal expressions by categories provides a much more interesting insight into this particular collection. There are certain categories like "Mathematics" or "Computing" whose documents have very few temporal expressions. Others categories such as "Geography", "History", or "Media", on the other hand, have a high number of temporal expressions. A manual inspection of some of those categories confirms these numbers. As an example, a history document like `Tibet_during_the_Ming_Dynasty` (175 chronons in 14,653 tokens) is likely to have more references to time than one about food and drink like `Saffron` (61 chronons in 5,603 tokens).

Figure 4.11 shows the distribution of temporal expressions in categories 1 to 12. The x-axis represents the size of the document and the y-axis the number of temporal

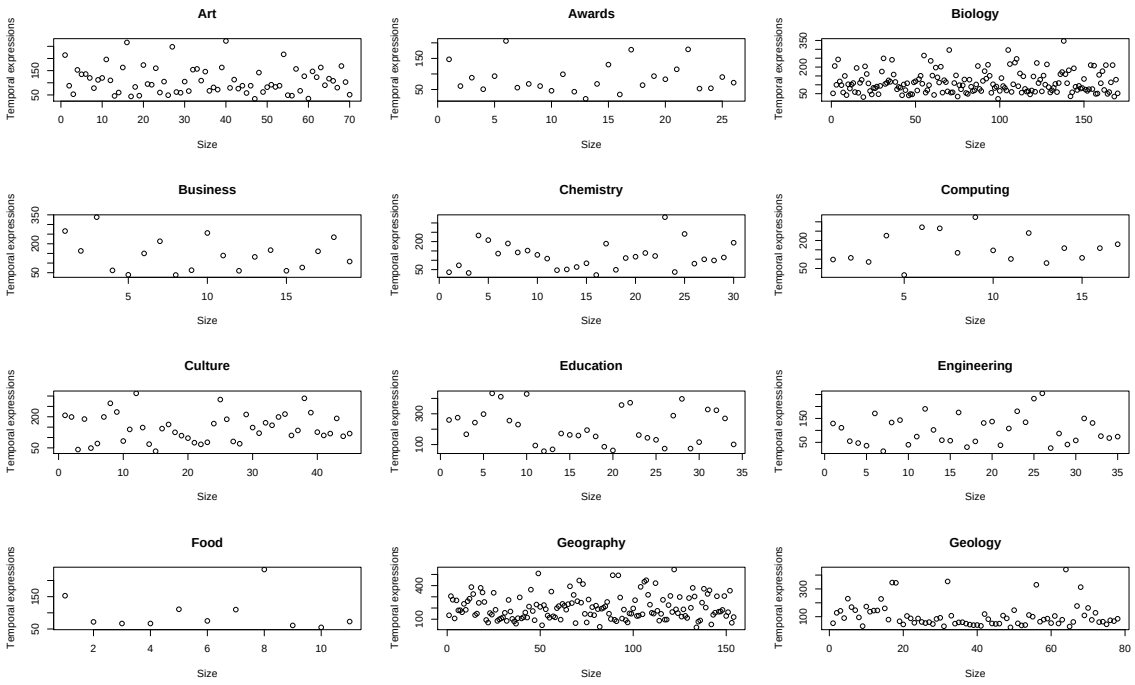


Figure 4.11: Distribution of temporal expressions for categories 1 to 12 from featured articles in Wikipedia.

expressions. Similarly Figure 4.12 shows categories 13 to 24 and Figure 4.13 shows the last four categories.

If we compute the ratio of temporal expressions over words or tokens per category, the results are very similar to TimeBank. In most of the categories one can find a temporal expression per 50 words (1/50). The exceptions are the categories Computing, Education, Geography, Geology, Media, Music, Sports, and Transport, where one can find a temporal expression per 33 words (3/100 or approximately 1/33).

In general, the approach described above can serve as an important step to first temporally analyze a collection as a whole before individual documents are explored. In particular, the user can be provided with such collection analysis results in order to get a good idea of the utility of subsequent document exploration tasks.

One can take another look at the same categories and see which section or part of a document contains the most number of temporal expressions. An approach is to split

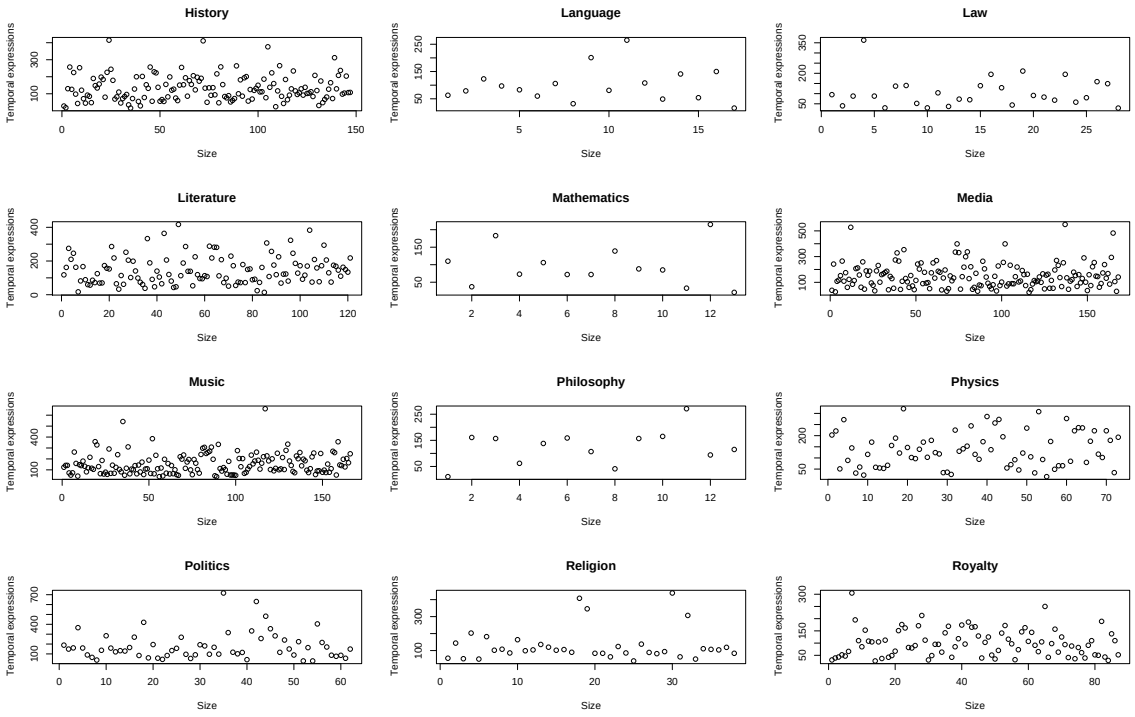


Figure 4.12: Distribution of temporal expressions for categories 13 to 24 from featured articles in Wikipedia.

the total number of sentences with temporal expressions in four parts (Q1 = 0-25% Q3 = 25-50%, Q3 = 50-75%, and Q4 = 75-100%) and select the maximum.

Figure 4.14 shows the distribution of temporal expressions according to their location in a document for categories 1 to 12. The x-axis represents the four quarters of a document and the y-axis the number of documents whose most frequent number of temporal expressions occur in a particular part of the document. Similarly Figure 4.15 shows categories 13 to 24 and figure 4.16 shows the last four categories.

The results vary from category to category. Overall, most of the allocation of temporal expressions in a document occurs in the first or last quarter of a document. For those documents that contain several references to time, it is likely that time information can be found early in the text. The explanation for the high percentage in the last quarter is due to the fact that most of Wikipedia articles contain a section on references or external links, which include mostly the year chronon.

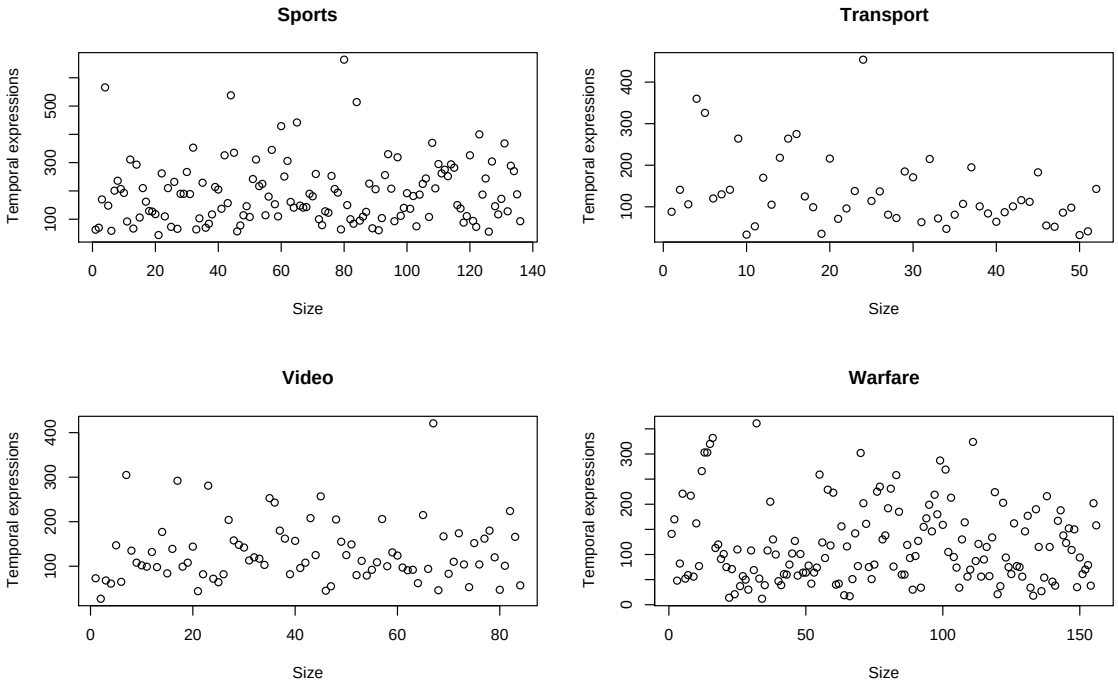


Figure 4.13: Distribution of temporal expressions for categories 25 to 28 from featured articles in Wikipedia.

4.5.3 File Names in Wikipedia

While analyzing the collection in the previous experiment, we noticed that page names in Wikipedia are also very descriptive of their content and in some cases, they do contain time information. Furthermore, a visual inspection reveals that the categorization of temporal expressions introduced in the previous chapter can be applied to file names (or rather titles of pages) well. Some examples of file names that contain temporal expressions:

```
1928_Okeechobee_Hurricane
1933_Atlantic_hurricane_season
1980_eruption_of_Mount_St
1983_Atlantic_hurricane_season
History_of_Test_cricket_from_1877_to_1883
Jersey_Shore_shark_attacks_of_1916
Yesterday_(song)
Glorious_First_of_June
November_(film)
```

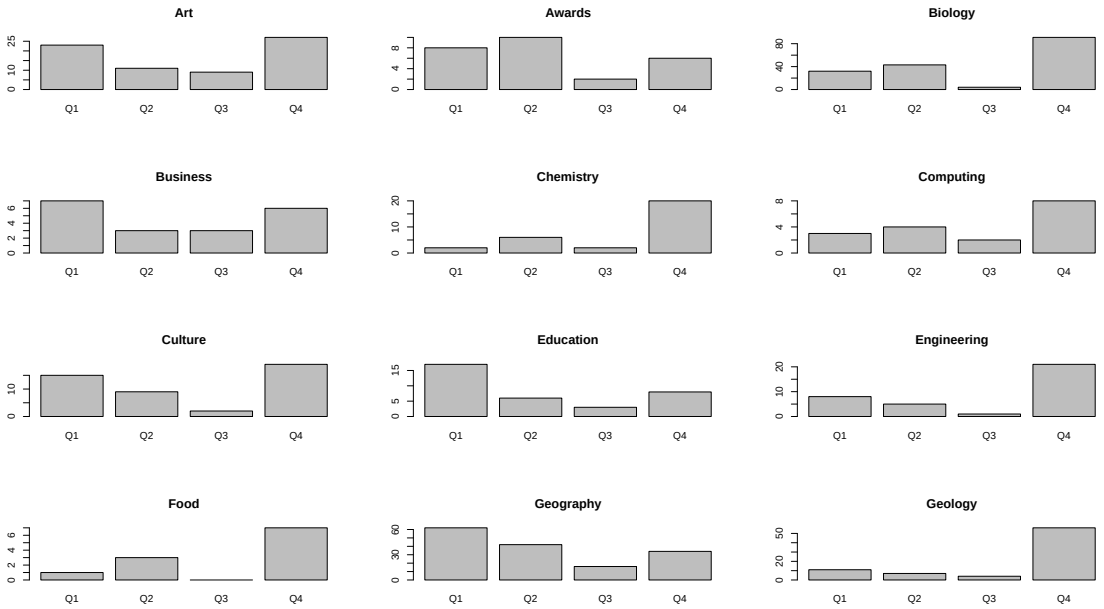


Figure 4.14: Bar plots for temporal expressions according their location in each of the four quarters of a document. This graph shows categories 1 to 12 from featured articles in Wikipedia.

We can also process a file name as a short fragment of text using the same infrastructure that we applied for processing whole documents. We only need to produce short documents given the file names. A straightforward approach is to eliminate join characters (e.g., "_", "(", ")", etc.) and we have the desired document. For example, the file name `History_of_Test_cricket_from_1877_to_1883` can be treated as the document `History of Test cricket from 1877 to 1883`, which contains 2 temporal expressions.

Of the 2,050 featured documents, 76 documents have 1 or more temporal expressions. From those 76, 3 have 2 temporal expressions in the file name. These are 3% of the collection, which looks somewhat low given the fact that a manual inspection of the file names results in a higher percentage (7%).

By looking at some file names that do contain the year chronon, like `1999_Sydney_hailstorm`, the temporal tagger is unable to detect the correct chronon (1999) from the text fragment `1999 Sydney hailstorm`. Now, if we reverse the content to `Sydney hailstorm 1999`, it does. This shows that NLP techniques are not perfect and in some cases, like this

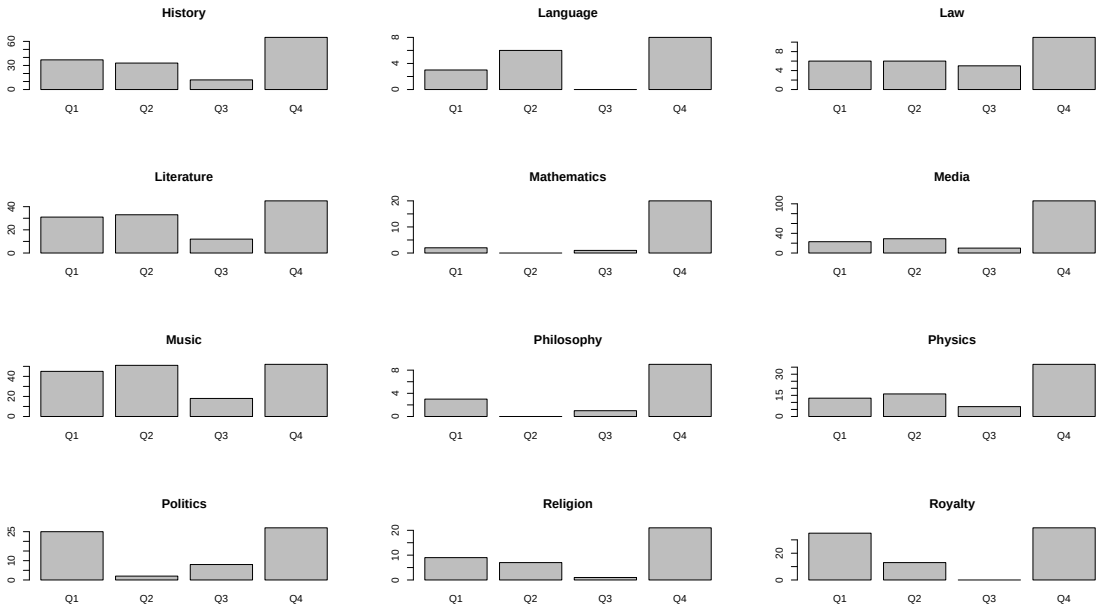


Figure 4.15: Bar plots for temporal expressions according their location in each of the four quarters of a document. This graph shows categories 13 to 24 from featured articles in Wikipedia.

one, could result in incorrect answers. Nevertheless, the experiment showed the flexibility of the temporal annotation pipeline.

In summary, for those document collections where the filenames are not a sequence number (like `doc12345abc.txt`) and that are likely specified by their editor, it is also possible to find the three types of temporal expressions (explicit, implicit, and relative) as one should find in a given document.

4.6 Related Work

To our knowledge there is little work on analyzing and organizing documents based on their temporal information. Time information as a feature beyond the traditional timestamp notion has not been applied to dissect a document.

A possible explanation is that most of the existing test collections are about news archives or similar variations that, as we showed in the experiments, do not contain a lot

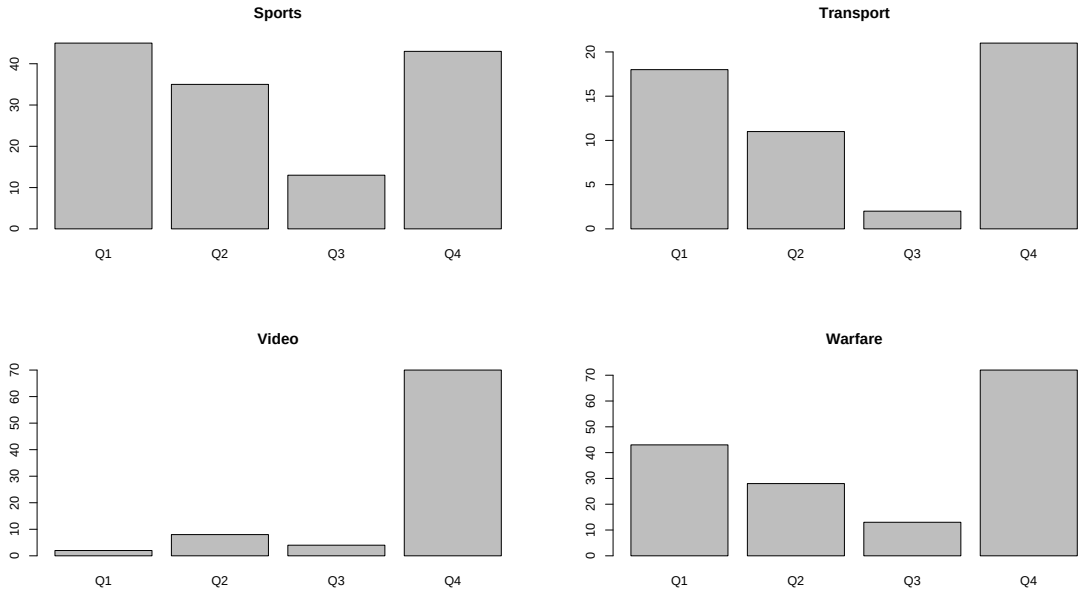


Figure 4.16: Bar plots for temporal expressions according their location in each of the four quarters in a document. This graph shows categories 25 to 28 from featured articles in Wikipedia.

of temporal expressions. The time frames project is one approach to augment news articles by extracting time information [61]. Extensions to document operations like comparing temporal similarity of two documents in the context of news articles is presented in [65] (see also [58]). Temporal mining of blogs is presented by Qamra *et al.* in [85]. Recently, new research has also emerged for future retrieval [16] where temporal information can be used for searching the future.

In other domains like medical informatics, documents that contain temporal information are very important. For example, discharge summaries provide a rich history of events for a patience. Modeling discharge summaries as a temporal constraint satisfaction problem is presented by Hripcsak *et al.* in [52]. Finding temporal order in discharge summaries is described by Bramsen *et al.* in [20].

Query log analysis is a very active area of research for identifying user intent. Recently, there is emerging work by Nunes *et al.* on analyzing temporal expressions user queries, which are treated as short fragments of text for extraction purposes [79].

4.7 Summary

Temporal expressions appear in a document collection according to the classification that we have previously introduced. However, this is not sufficient to characterize documents based on temporal information when the goal is to further explore a corpora.

By defining quantifiable measures like temporal richness, specificity, and boundaries, we are in a much better position for performing an analysis based on temporal content. We presented how these measures can be computed after extracting temporal expressions from a document and also studied their distribution and continuity. We defined the concepts of temporal coverage and order that we later use for computing a temporal similarity measure.

All those temporal measures can be used in different cases and the value they provide is that they reflect different temporal aspects of a document collection.

We investigated where temporal expressions occur in the whole document and the sentence level. References to time tend to appear early in a document and then it varies depending on the content. For those documents that contain lots of citations or references, there is quite a bit of temporal expressions at the last sections and usually at end of sentences.

We performed a number of experiments using case studies that showed the results of our analysis using the concepts introduced. The first experiment based on the TimeBank corpus, which covers news articles. The experiment shows that news article do contain temporal information that is, usually, within the scope of the particular news. The articles are usually short in terms of size, and in some cases the only temporal expression is the document timestamp.

The second experiment consisted of using a high quality subset of Wikipedia called featured articles. The analysis shows that documents in certain categories have an abundant number of temporal expressions. We also showed in which part of a document most of the temporal expressions appear. For this data set, most of the temporal expressions are located in the early sections of the documents or at the very end. These type of studies can provide

quantitative metrics to evaluate if a document collection has richer temporal content for exploration purposes.

We also use the same Wikipedia data set to study the distribution of temporal expressions in file names. Because of the usage of Wiki technology, a title name in a Wikipedia page is the same as in the correspondent file name. The analysis shows that, for Wikipedia pages that are edited by people, temporal expressions also occur in certain file names.

Chapter 5

Time-based Document Ranking and Clustering

In the previous chapter, we defined a number of temporal measures that can be applied to document collections to get a better understanding of their temporal characteristics.

One can think of those metrics as a static view of a document collection. There is no user involved in the analysis process. To complement our research with a dynamic view, we study how a search application can benefit from temporal information for retrieval purposes. In this chapter, we introduce a time-based document clustering technique that allows for the grouping of search results timelines. We also define a temporal document snippet feature that can be used in combination with the clustering approach, for presenting a short text fragment that is temporally aware.

This chapter is organized as follows. The next section describes motivation and objectives. Section 5.2 presents our temporal clustering approach. We explore different query scenarios in Section 5.3. We describe our temporal document snippet technique in Section 5.4. Section 5.5 outlines the implementation of a prototype. In Section 5.6 we demonstrate the evaluation framework and experimental results. The temporal snippet evaluation is presented in Section 5.7. Finally, Section 5.8 presents related work.

5.1 Motivation and Objectives

Search result clustering is a feature integrated in some of today's search engines, allowing users to further explore search results. However, only little work has been done on exploiting temporal information embedded in documents for the presentation, clustering, and exploration of search results along well-defined timelines.

In this chapter, we detail how a given collection of documents with temporal document profiles is organized along a multiple-granularity timeline in the form of document clusters. This approach can be applied to either an unranked collection of documents or documents in a search results list. However, in the following, we will focus on documents in a hit list of search results.

The time-based clustering algorithm, called *TCluster*, described in the next section is realized as add-on to any standard document retrieval technique. It can be invoked automatically by the retrieval technique or by the user, if he/she desires a time-based exploration of retrieval results.

We also include in this chapter, a technique for constructing a document snippet that leverages temporal information. This snippet can be used as preview that outlines the main events in a document.

To summarize, the objectives of this chapter are to:

(i) Introduce a new time-based clustering algorithm.

(ii) Investigate how such a clustering approach can be useful for search and exploration.

(iii) Introduce a temporal-based document snippet algorithm.

(iv) Show a prototype implementation of TCluster using existing technologies.

(v) Study query scenarios and its effect in temporal ranking.

(vi) Present an evaluation framework and experiment results.

5.2 TCluster

We assume that for a query term q against a document collection $\mathcal{D}$, the retrieval algorithm determines a hit list $L_q = [d_1, d_2, \ldots, d_k]$ of k documents. We also assume that all documents in $\mathcal{D}$ have a unique id and that there is an inverted index file that allows to return a set of documents for a given query term or temporal expression. Given such a hit list, the temporal document profiles are used to construct a *time outline* for the documents first. The documents are then clustered along this timeline, again based on their document profiles. A single cluster can be thought of as a bin that contains only documents with temporal expressions matching the cluster label, which corresponds to some chronon. The organization of clusters along a timeline as well as the lattice structure imposed among timelines then allows for the exploration of document clusters at different levels of time granularity.

The following sections elaborate on the individual steps of the TCluster algorithm.

5.2.1 Constructing a Time Outline

The first step in organizing documents along a multiple-granularity timeline is to construct a *time outline* for the documents in the hit list L_q. For this, all chronons are extracted from the temporal document profiles of the documents in L_q. Given a document profile $tpd(d)$ for document, we denote this multi-set of chronons $ch(L_q)$, defined as follows:

$$ch(L_q) := \{\{c \mid d \in L_q \text{ and } (e, c, p) \in tdp(d)\}\}$$

Note that the elements in $ch(L_q)$ may come from different timelines. More importantly, among the elements in $ch(L_q)$ there are (not necessarily unique) minimum and maximum chronons, which describe the *lower* and *upper bound* of the time outline for the documents in L_q. Based on the range between lower and upper bound, a time granularity for the time outline is chosen.

Assume, for example, L_q contains a document with a temporal expression mapped

to the year 1974 (as lower bound) and another document with a temporal expression mapped to the year 2007 (as upper bound). Then the temporal range of L_q is several years, and the type T_y is chosen as time outline for L_q. On the other hand, if L_q contains documents, say some news articles, whose temporal range is only a few weeks, then the type T_w would be more appropriate as time outline. In general, a time outline is a timeline representation that describes the *temporal range of documents* in L_q, independent of the "temporal distribution" of documents along this timeline. We will elaborate more on the distribution aspect below. Without loss of generality, in the following, we assume a time outline based on the timeline T_y.

5.2.2 Document Clustering

In the next step of the TCluster algorithm, the timeline chosen as time outline for L_q is used to *normalize* the chronons in $ch(L_q)$, here according to T_y. That is, if a chronon c in $ch(L_q)$ is of a granule finer than year, only the year component of c is used. In general, we denote such a type of normalization of a chronon c based on a time granule g ($g \in \{y, m, w, d\}$) as $norm_g(c)$. For example, $norm_y$("15/4/1966") = "1966", and $norm_m$("15/4/1996") = "4/1966". The *labels* for the initial (most coarse-grained) document clusters for L_q and time granule g are then determined by the following set.

$$ch_g(L_q) := \{norm_g(c) \mid d \in L_q \text{ and } (e, c, p) \in tdp(d)\}$$

Intuitively, for the timeline T_y, $ch_y(L_q)$ simply contains a set of years such that there is at least one document $d \in L_q$ that has this year (as part of) its chronons. Assume there are l cluster labels $y_1, y_2, \ldots, y_l$, $y_j \in T_y$ in $ch_y(L_q)$ among which the precedence relationship $\succ$ holds. The documents in a cluster y_j, denoted $cluster(y_j)$, are then determined as follows:

$$cluster(y_j) := \{d \mid d \in L_q, \exists c : (e, c, p) \in tdp(d) \text{ such that } norm_y(c) = y_j\}$$

Obviously, if a document $d \in L_q$ has several components in its temporal document profile $tdp(d)$, it can be in several document clusters. This is intuitive, as in the text of d, there can be many explicit, implicit, and relative references to different points in time. The question then, however, is whether there is a *main cluster* for each document in L_q. Using temporal measures introduced in the previous chapter, such a cluster can easily be determined based on the distribution of the chronons in $tdp(d)$ with respect to the temporal range of the hit list L_q. For example, if the chronons associated with d refer to n different years, the main cluster for d, denoted $c\_main(d)$, would be the year for which d has the most chronons.

Finally, there can be a wide variation among the number of documents in each cluster. Some clusters might only have a very few documents whereas others have many documents, perhaps representing some kind of *hot spots*. For some years in T_y there might also be no documents at all that refer to that year. In general, from a user interface and document exploration point of view, if the time outline for L_q is represented to the user, each cluster is not only labeled by a year chronon but also shows information about the number of documents in that cluster. Referring back to the example of a topic search in a corpus of technical documents mentioned in the introduction, the cluster(s) with the most documents would then present the years in which the topic was "hot".

5.2.3 Ranking Documents in a Cluster

Thus far, the TCluster algorithm only determined the sets of documents belonging to each cluster along a timeline, here T_y. Clearly, documents in a cluster should be ranked to reflect the relevance of documents in $cluster(y_j)$ with respect to *both* the cluster label y_j *and* query terms q. While the ranking of documents in L_q is solely based on q, the ranking of documents in $cluster(y_j)$ now also takes the cluster label y_j and thus time into account. Key to such a ranking is the *distance* of the query terms q to the temporal expressions in the documents in $cluster(y_j)$.

Assume a document $d \in cluster(y_j)$ with temporal document profile $tdp(d)$. Let

$match_e(d, y_j)$ denote the number of times the query term q occurs *together* with an explicit temporal expression e, $(e, c, p) \in tdp(d)$, *in a sentence* in d such that $norm_y(c) = y_j$. Analogously, let denote $match_i(d, y_j)$ and $match_r(d, y_j)$ denote the number of matches with respect to implicit and relative temporal expressions in $tdp(d)$, respectively.

It is clear from the description of the *match* functions that the more often q occurs together with explicit temporal expressions matching y_j in d's sentences, the more relevant document d is in that cluster. Furthermore, there are several reasonable choices for a ranking that combines $match_e$, $match_i$, and $match_r$. For the TCluster algorithm, we choose the following function:

$$rank(d, y_j) := match_e(d, y_j) + \delta_i * match_i(d, y_j) + \delta_r * match_r(d, y_j), \quad \delta_i, \delta_r \in [0, 1]$$

That is, the sentence level co-occurrence of q with an implicit or relative temporal expression matching y_i can be weighted less than a respective co-occurrence with an explicit temporal expression. In order to deal with scenarios in which no single temporal expression in d matches y_j, we simply assume that $rank(d, y_j) = \delta_i$, because every document has at least a document timestamp as temporal expression (although this does not occur with q). After running multiple experiments with different temporal taggers, we believe that good detection of explicit temporal expressions is very desirable in such tools.

The above ranking function obviously leaves much room for further investigations. For example, the *match* functions could be extended to not only look at sentences in which the query term q occurs but also close-by sentences. Also, as implicit temporal expressions typically contain at least a year expression (plus the name of some event), even without a time ontology to match the exact day or period of that event, the year can be determined with high confidence; therefore, δ_i is typically close to 1. The choice of the δ_r heavily depends on the capabilities of the temporal tagger, in particular its ability to determine and correctly anchor relative temporal expressions.

Based on the above discussion, we use the following ranking approach in our TCluster algorithm. Given two documents $d, d' \in cluster(y_j)$, d is ranked higher than d' in $cluster(y_j)$, denoted $d >_{y_j} d'$, if either of the following conditions hold:

1. $rank(d, y_j) > rank(d', y_j)$.

2. $rank(d, y_j) = rank(d', y_j)$ and d is ranked higher in L_q than d'.

It should be noted that due to such a "time-based" ranking in $cluster(y_j)$, a document d can be ranked higher than a document d' in $cluster(y_j)$ although d' is ranked higher than d in the original hit list L_q. There are several obvious scenarios that make this point more clear. For example, if in document d' the query q occurs more frequently than in d, then d' is ranked higher in L_q than d. But if only in d the query q occurs together with the chronon y_j, then there is a closer relationship between q and y_j in d than in d'. This "re-ranking" is an important property of the TCluster algorithm and reflects the algorithm's focus on temporal information extracted from documents and represented in temporal document profiles.

5.2.4 Cluster Exploration

The construction of the time outline and the time-based clustering of hit list documents can be invoked by the user after after a standard document retrieval and ranking approach in an exploratory search interface (see subsection 5.5.1). For example, it can be applied to the hit list provided by a search engine. The document clusters organized along a timeline representing the temporal range of the documents and the ranked documents within each cluster now serve as the basis for a *user-driven, time-based cluster exploration*.

As discussed in Section 3.4, the composition of time granules leads to a lattice structure among timelines. This lattice structure is employed when the user wants to explore a given document cluster. The exploration typically starts with the timeline constructed in the first step of the TCluster algorithm, here clusters with year chronon labels $cluster(y_j)$, and occurs in a "drill-down" fashion.

Assume a cluster $cluster(y_j)$ with ranked documents $[d_1, d_2, \ldots, d_l] \subseteq L_q$. The cluster can be *refined* based on a timeline T with $T_y \gg T$, i.e., T can be T_m or T_d. That is, the chronon is expanded to its constituent granules, providing the user with a drill-down operation.

Assume the user chooses the timeline T_m to refine a particular $cluster(y_j)$. The TCluster algorithm then first constructs a (now more fine-grained) time outline for this particular cluster and its documents, based on months. Then, the documents in $cluster(y_j)$ are partitioned into clusters labeled by month/year chronons, and ranked in respective clusters. These two tasks follow the same procedures as described in subsections 5.2.1 and 5.2.2. The only difference now is that in the clustering step, the chronons associated with documents in $cluster(y_j)$ are normalized by the granule month and not year anymore, thus leading to a different set of cluster labels, namely months in the given year y_j. The ranking of the documents in the resulting month-based clusters m_k $(1 \leq k \leq 12)$ is done in the same fashion as described in Section 5.2.3. Choosing finer and finer timelines clearly leads to an effective way to explore and compare document clusters and the ranking of documents in clusters, all functions that can effectively supported by a simple user interface, as described in the following.

5.3 Query Scenarios

One can now explore search scenarios where the user employs the clustering functionality to explore and retrieve document that can potentially satisfy an information need.

Detecting the presence of temporal expressions in a query can be implemented by extending the query language or in a user interface. Extending a query language to add temporal semantics is out of the scope of this research. Instead we provide some features as part of our exploratory search prototype that we describe in the next chapter.

As example, say that the user is interested in biographies about famous researchers in physics from a document collection like Wikipedia. A query on [Einstein] retrieves documents about Albert Einstein organized in a timeline according to TCluster. The user

can explore the timeline and expand/contract the appropriate clusters if needed. In this scenario the query does not contain temporal expressions so there is no information about time that we can derive from the input string.

Following the same example, the user may be interested in a particular year and formulates the query [Einstein 1921]. Here there is an explicit temporal expression in the query so the engine returns those relevant documents that are richer, specific, and contain such chronon. The user can also re-formulate the query with a particular time period in mind like the query [Einstein 1921-1939]. The engine then returns document that cover those chronons.

5.4 Temporal Snippet

TCluster groups relevant documents according to the user query in a timeline. When the user selects a particular chronon, the engine presents the titles and URLs for that set of documents. Because the title and URL can sometimes be misleading or incomplete, a traditional search engine would also show a snippet or a small document summary that contains a bit more information about the document.

In the early days of the Web, a summary consisted on the first 80 characters of a document. Nowadays modern engines use *query-biased summaries* or variations of *keyword-in-context* (KWIC). Those summaries or sentence fragments contain query terms that are extracted from the document and are presented in the context in which the query terms appear. On the contrary to these query dependent summaries, a well known approach is to provide a *generic summary*, which is a short fragment that represents the key points in a document.

One can argue that a document snippet that leverages temporal information would be an interesting alternative. Intuitively, it makes sense to include time in a snippet. For example, a well known fact about Einstein is that he won the 1921 Nobel Prize in Physics. We define a *temporal snippet* as a short fragment of text that contains two or three sentences with the most relevant chronons in chronological order. How can one use all the temporal

information that is available in a document to select relevant sentences that can be used to construct such a snippet?

We assume that for a document d from a document collection $\mathcal{D}$, the temporal snippet algorithm determines the top n ($1 \leq \mathrm{n} \leq \mathrm{m}$) sentences from a list $S_d = [s_1, s_2, \ldots, s_m]$ of m sentences in the document. The algorithm *TSnippet* contains two parts: sentence selection and ranking.

5.4.1 Sentence Candidate Selection

The most obvious attempt for generating a temporal snippet is to select the first sentence that contains a temporal expression that is not a timestamp.

Surprisingly, the results for this näive approach are not that bad for documents with good structure. In the examples of the CNN 9/11 news article (`September 11. Chronology of terror`) and Einstein page in Wikipedia (`Albert Einstein (March 14, 1879 - April 18, 1955) was a German physicist ...`), the results are good from the standpoint that they reflect a valid event in the document. Unfortunately, this simple approach cannot be generalized for all types of documents due to the following problems:

(i) Photo captions: a document may include a figure or photo in the first couple of sentences, like `Einstein in 1947`.

(ii) Very long sentences: if the sentence detector fails to identify in the document an object like a table, it would produce a very long sentence that contains every cell item. This is not semantically correct and in the event of a chronon in a cell, this incorrect sentence is selected as snippet.

(iii) The first sentence does not contain an explicit temporal expression. For example, one of the first sentences in the document `Doctor_Who` is `Current Doctor Who title sequence ....` This is a valid sentence but not very useful as a snippet.

We define the following criterion for selecting candidate sentences:

(i) A sentence must contain at least one temporal expression.

(ii) All temporal expressions in the sentence are explicit.

(iii) Sentence length is $\geq$ *sentence-min* and $\leq$ *sentence-max*.

The values of *sentence-min* and *sentence-max* should be adjusted according to a particular document collection. In our case we set the values to 7 and 28 respectively based on sentence lengths statistics from newswire text in Manning and Schütze [70].

What is now needed is a ranking function that, given a number of candidate sentences with temporal expressions, would produce a sorted list of relevant sentences.

5.4.2 Ranking

How can one define an arbitrary ranking function for a set of sentences? One approach would be to use the sentence length (or standard deviation) and a low ratio of position/length as parameters. Among other solutions, one is to select the sentence that contains the most popular chronon and is also the first one that contains such chronon in the document. Another alternative would be to include the frequency of chronons in the text and within the sentence.

A much better approach is to identify which features are interesting and define a ranking function that combines them. In our case, we select the following features:

(i) p: position of the temporal expression within the sentence (ratio of position/sentence length).

(ii) s: sentence number.

(iii) sl: sentence length in tokens.

(iv) co: chronon appearance order (e.g., first, second, etc.).

(v) cf: chronon frequency in document.

(vi) cfs: chronon frequency in sentence.

The ranking function is a linear feature combination where each parameter is tunable:

$$rank(d) := \alpha p + \beta s + \gamma sl + \delta co + \epsilon cf + \varepsilon cfs$$

The above ranking function is query independent. In other words, no matter what the query is, the snippet is the same. Making it query dependant is a matter of adding a similarity measure to the overall ranking function so it does take into account the presence of terms in the final result. We add the cosine similarity feature introduced in subsection 2.1.1 in the linear combination. The new rank function is then:

$$rank(d, q) := \alpha p + \beta s + \gamma sl + \delta co + \epsilon cf + \varepsilon cfs + cos(d, q)$$

The TSnippet algorithm is then:

1. Segment the document d in sentences $s_1, \ldots, s_n$ and treat each sentence as a separate document.
2. Select candidate sentences according to sentence selection criteria.
3. Apply rank function to selected sentences.
4. Select top-n sentences sorted by chronon as temporal snippet.

The following is the temporal snippet (with number of sentences $n = 2$) for the Einstein page in Wikipedia for the query [Einstein]. The sentences are presented in chronological order.

```
In 1905, while he was working in the patent office, Einstein
had four papers published in the Annalen der Physik, the leading
German physics journal.

Einstein received the 1921 Nobel Prize in Physics for his services
to Theoretical Physics, and especially for his discovery of the
law of the photoelectric effect.
```

5.5 Prototype Implementation

In our prototype, we use a combination of existing technologies to demonstrate the feasibility of our clustering approach. The prototype basically realizes two main components: 1) a back-end, which supports corpora processing and storage as well as index creation, and b) processing unit, which realizes query processing and time-line construction and also implements the user interface to the system.

The back-end processing is performed using the same set tools as described in Section 3.9. For query processing, the system takes some user query terms as input, executes the query, i.e., it performs some document ranking, and runs the TCluster algorithm on the search result hit list. The search results are then presented along a base timeline in a user interface. The presented timeline is composed of clusters (labeled by year chronons) and provides the user with means for a more fine-grained clusters exploration.

5.5.1 User Interface

The user interface to our prototypical system is realized as a Web-based user interface (see also Figure 5.1). In the interface, a user can enter query terms and explore the resulting document clusters returned for a query. Using a Java middle-ware layer, query terms are submitted to the Oracle database, which uses a standard tf-idf ranking scheme to first determine the hit list L_q for a query q. This type of standard ranking is realized through some simple PL/SQL procedures and functions in the database back-end.

The TCluster algorithm described in Section 5.2.4 is realized as PL/SQL components as well. If the documents in L_q do not contain any temporal expression, the document timestamp is used as the only component of the temporal document profile for our clustering technique. In the prototype, the year timeline is used as base timeline for the clusters determined in 5.2.1. Using the *match* functions, the implementation of TCluster also determines a ranking of the documents in each cluster in 5.2.2. In our implementation we set $\delta_i = 0.8$ and $\delta_r = 0.2$. As discussed in Section 5.2.4, for this, the sentence level distance between the query and temporal expressions is set to 7 (default value for *sentence-min*).

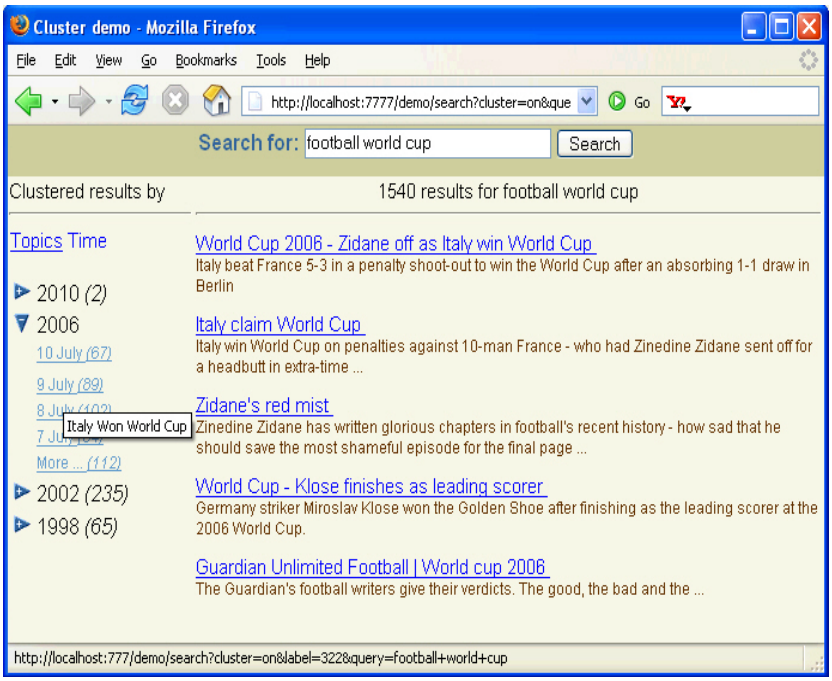


Figure 5.1: Timeline cluster for results for query [football world cup].

Information about the base-line document clusters is sent back to the user interface client and displayed along a timeline in descending year order. As can be seen in Figure 5.1, for each cluster (labeled with a year chronon), the number of documents in the cluster is shown. A cluster can be expanded, allowing the user to explore the documents in a cluster based on more fine-grained timelines, according to 5.2.3.

5.6 TCluster Evaluation and Results

The primary objective of our evaluation here is to show the clear advantage our temporal clustering approach has over existing document clustering approach that only perform document clustering on document timestamps. We also aim to show that clustering results determined by our approach are indeed more meaningful than such standard clusters and also allow for flexible document and cluster exploration based on different timelines and time granules, respectively.

5.6.1 Evaluation Guidelines

There has been little work published on evaluation for automatically generated timelines. We adopt some of the evaluation guidelines for automated testing that are outlined on Cronopath [23]. There are two important features of a timeline that we are interested in: *precision* and *presentation*.

Precision is defined as the fraction of the retrieved documents that are relevant, as was defined in subsection 2.1.3. The main point here is that all relevant documents must be included in the timeline. The timeline should also contain appropriate labels for each time granule.

Presentation involves displaying the timeline in a graphical form. The most intuitive way is to draw a line and mark the main time granules from left to right in ascending order, thus forming a time outline. There are other more advanced designs for visualizing timelines, which can be useful for exploratory search and will be discussed in Chapter 6. We did not perform an extensive user study for the evaluation of the timeline presentation feature. Rather, we concentrate on the precision, because it demonstrates the underlying concepts of timeline construction and search result clustering in a better way.

We conducted two sets of experiments with our time-based document clustering system. The first set of experiments uses part of the well-known Internet directory DMOZ [35]. The second set of experiments uses a document collection in which temporal expression have been manually annotated.

5.6.2 DMOZ

DMOZ is a multilingual open content directory, which is constructed and maintained by a community of volunteers. For the experiment, we took the DMOZ category "World Cup" in the larger category football. The "World Cup" category contains a collection of articles that have been put into context by users, meaning that each document corresponds to either of the World Cups in 2010, 2006, 2002, 1998 and 1994, leading to a user-specified collection of five document clusters. We used a crawler to obtain all doc-

uments in this category (not including those with broken links or non-English content), and applied the document processing pipeline presented in Section 3.9 to the crawled 120 documents. We then applied our TCluster algorithm to compare the number of clusters obtained through TCluster with the 5 pre-defined clusters in this category.

Query	**DMOZ**	**TCluster**
World Cup	5	21

Table 5.1: Precision of the timeline.

Precision of the timeline. The number of manual categories in the DMOZ World Cup category is lower than the number of clusters determined by TCluster. DMOZ has all events since 1994 while TCluster identifies many more past tournaments.

While this DMOZ entry has only five categories that represent the past four tournaments and the forthcoming one, TCluster determined 16 more clusters (see Table 5.1). Why this discrepancy? It is clear that each World Cup document has a single event as the main theme. Thus, documents are well classified by users in terms of the actual event. On the other hand, the same document usually refers to past events and winners. Therefore, it is also possible to view the same document content along the time line of the history of the World Cup, meaning that a single document can be relevant to more than just one single event.

5.6.3 TimeBank

The second set of experiments uses the TimeBank 1.2 corpus [101]. The objective of the experiments was to show that using temporal expressions, if correctly identified and made readily accessible to TCluster, can significantly improve the precision and granularity of timelines along which documents are clustered. For comparison purposes, we run a set of search queries against this annotated TimeBank document collection and the corresponding collection that had no such annotations and marked-up temporal expressions, respectively. For the latter "plain text document collection", only the document timestamp has been

used for the TCluster algorithm.

Intuitively, for the plain text collection, documents in the search result are clustered based on their document timestamp. Applying TCluster to the plain text documents should at least result in the same number of clusters, as TCluster considers the document timestamp as part of a temporal document profile. However, if there are more temporal expressions accessible to TCluster in a temporal document profile, the resulting timeline along which documents are clustered should be more precise by 1) providing more clusters along the timeline, and 2) providing more documents in a cluster, as a document then can belong to more than one cluster. Table 5.2 shows the results of this experiment using TimeBank.

Query Term	**Without Annotations**	**With Annotation**
stocks	2	2
president	4	6
kalashnikov	1	1
internet	1	1
columbia	2	4
financial	2	2
new york	2	3
dow jones	2	3
mccaw	1	3
ireland	1	2

Table 5.2: Number of clusters for the same document collection without and with marked-up temporal expressions.

The more temporal expressions are made explicit (besides just the document timestamp), the better the precision of the document clustering.

Let us take a closer look at one particular query that asks for documents in the TimeBank collection that are relevant to a company named “McCaw”. For the plain text collection, TCluster only returns one cluster with the label 1989. On the other hand, for the annotated collection, TCluster returns three clusters with cluster labels 1984, 1989, and 1994. This means that there is at least one document relevant to the query [McCaw] that has three different temporal expressions in it. The following is made more evident

by a fragment of the document with the sentences that contain the term McCaw near to temporal expressions.

```
<s>
 In effect, McCaw has forced ...
 for the stake
 <TIMEX3 ... value="1989-10" ...>earlier this month
 </TIMEX3>
</s>... <s>
 Only McCaw's proposal requires the company
 to begin an auction ...
 <TIMEX3 ... value="1994-06" ...>June 1994</TIMEX3>
 for remaining shares at third-party prices.
</s>... <s>
 Mr. Leon of Bear Stearns speculated ...
 <TIMEX3 ... value="1984" ...>1984</TIMEX3>
 </s>
```

As one can see, there are three sentences that have `TIMEX3` tags with different values. Thus, the corresponding temporal document profile not only contains the document timestamp. This shows that using temporal expressions that have been derived and made explicit in temporal document profiles significantly help identify information about this particular company with respect to three different time chronons.

Finally, we also ran a few queries to get a better understanding of the limited use of temporal expressions in well-known clustering engines such as Vivísimo. As our prototype does not have such a deep coverage in terms of accessible document collections, here we present some anecdotal evidence of the under-utilization of temporal information in these engines. The task consisted of counting temporal expression frequencies over the total number of temporal expressions in labels on a subset of sports news. Using variations of the query [football world cup], we are able to report that the percentage of temporal expressions in the top-10 hit list is around 10%. TCluster was 35% higher, because the labels are temporal expressions. This is, of course, an unfair comparison and the only point is to suggest that temporal expressions as cluster labels in those Internet systems are rare.

5.7 TSnippet Evaluation and Results

The objective of the temporal snippet evaluation is to show that recognizing temporal information can help in creating a different snippet that is more precise with respect to time. Current search engine snippets are too short and recent studies suggest that a longer summary might be desirable [57]. TSnippet does include full sentences, not just fragments.

5.7.1 Experimental Setup

The experiment consists of measuring the precision of a snippet (temporally speaking) using the same subset of Wikipedia articles used in the previous Chapter. We are not interested in measuring recall because all documents have temporal expressions, as we have demonstrated in Section 4.5.

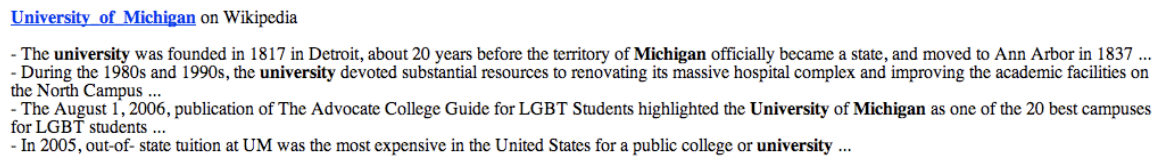
For tuning the ranking function described in Section 5.4, we initially experimented with a snippet of four sentences ($n = 4$), mainly for debugging purposes. Intuitively, we are interested in sentences that contain a frequent chronon that appears in the first few terms with a high cosine similarity score. As expected, selecting a good weighting scheme for each parameter is a combination of science and art.

After experimenting with different parameters and values, we set $\alpha = 1 - p$, $\beta = 1/s$, $\gamma = 0.25$, $\delta = 1/(\log(co) + 2)$, $\epsilon = 0.02$, and $\varepsilon = 0.01$. It is important to note that our goal here is not to define the perfect ranking function. The objective is to define a function that provides good quality temporal snippets for a user.

5.7.2 Featured Articles in Wikipedia

For this experiment, we have developed an evaluation database program that takes each document from the data set and extracts the appropriate snippets from TSnippet, Google, and Yahoo!. For example, the document `University_of_Michigan` is transformed into a query [University of Michigan] for TSnippet. In the case of the Web search engines, the query is restricted to the Wikipedia domain [University of Michigan site:.en.wikipedia.org].

We use sentence number $n = 2$ for this experiment. Sentence length was not part of this evaluation. Figure 5.2 shows the temporal snippet for the query [University of Michigan] for $n = 4$.


University_of_Michigan on Wikipedia

- The **university** was founded in 1817 in Detroit, about 20 years before the territory of **Michigan** officially became a state, and moved to Ann Arbor in 1837 ...
- During the 1980s and 1990s, the **university** devoted substantial resources to renovating its massive hospital complex and improving the academic facilities on the North Campus ...
- The August 1, 2006, publication of The Advocate College Guide for LGBT Students highlighted the **University** of **Michigan** as one of the 20 best campuses for LGBT students ...
- In 2005, out-of- state tuition at UM was the most expensive in the United States for a public college or **university** ...

Figure 5.2: Temporal snippet for University of Michigan page in Wikipedia.

The snippet results are presented on the screen for evaluation. The evaluator sets time=yes if the snippet contains temporal information, otherwise times=no. If the snippet contains any of the query terms highlighted, the evaluator sets kwic=yes, otherwise kwic=no. A snippet that contains a URL with a highlighted term is not considered a proper snippet.

Once the evaluation of each document is completed, the precision is computed. Table 5.3 shows the results of the evaluation of 25% of the document collection. A (t) in a column indicates that temporal information is available in the snippet. A (k) in a column indicates that the snippet contains query highlighted keywords.

TSnippet (t)	**TSnippet (k)**	**Google (t)**	**Google (k)**	**Yahoo (t)**	**Yahoo (k)**
0.984%	0.946%	0.428%	100%	0.273%	0.973%

Table 5.3: Snippet precision for TSnippet, Google, and Yahoo!

From Table 5.3, we can see that TSnippet is very precise for presenting snippets with temporal information. This is expected because the algorithm should detect sentences with temporal expressions. However, some sentences with relative expressions or incorrect expressions are appearing on top. For example, in the document `Star_Wars_Episode_IV:_A_New_Hope`, TSnippet selects the sentence `In response, 20th Century Fox ...` that contains an incorrect annotated expression (20^{th} Century) instead of a company name (20^{th} Century Fox). In terms of highlighted query terms, TSnippet is missing some documents.

One possible explanation is that sentences are truncated for displaying purposes, so if a query term appears at the end it is likely that it would not be selected.

Google has a perfect score for KWIC. The precision for presenting snippets with temporal expression is 43%. Yahoo! is doing poorly, with 27% of their snippets presenting temporal information. Also for Yahoo!, the score for KWIC is 97%, with some examples of missing highlighted keywords in the snippet or highlighting in the URL only.

Why do Google and Yahoo! have those numbers? Where is the discrepancy? Without knowing their implementation, one could guess that their snippet is constructed around text that is very similar to the title and URL. As we have mentioned before, snippets rely heavily on the query. So if the query is [1980 eruption Mount St. Helens site:.en.wikipedia.org], Google and Yahoo! will include the year chronon in the snippet because is part of the query and probably it can also be found in the title and URL. Now with a query with no chronon like [National emblem Belarus site:.en.wikipedia.org], the snippet does not contain temporal information for Google and Yahoo!. However, there are cases where the absence of a chronon in a query like [Aaron Sorkin site:.en.wikipedia.org], produces a snippet with temporal information. In this latter example, the temporal expression (a birthday) appears next to one of the highlighted terms.

So far in this experiment we have assumed that using the Wikipedia page title restricted to the Wikipedia domain as query, would return the desired document in the first position of the hit list. Unfortunately, Web page popularity can produce some unexpected results. For example, for the query [Cape Feare site:.en.wikipedia.org] that represents the document `Cape_Feare`, Web search engines rank `Cape_Fear_(1991_film)` higher on the hit list.

In summary, constructing a snippet that always contains temporal information is a novel approach. The experiment and evaluation showed that our proposed technique has very good precision for detecting snippets with time information.

5.8 Related Work

A technique related to our approach is hit list clustering. In general, hit list clustering groups search results into categories that are derived from the actual search [115]. Instead of processing the entire document set, hit list clustering uses a small document set that fits several well-known substring searching algorithms. Current hit list clustering engines like Vivísimo rely on a separate search engine that provides some information like Web page title, URL, and document snippets for the construction of the clusters [110]. Rarely a temporal expression appears as part of the cluster labels.

Using the Vivísimo search engine as baseline, there have been a number of approaches to improve hit list clustering, such as named entities for labeling clusters [103, 104, 86] or grouping by specific entities and citations [2].

A popular hit list clustering construction technique is based on suffix trees; alternatives to this technique are described by Ferragina and Gulli [39]. There is little work on exploiting temporal information associated with documents for clustering search results. An earlier version of our approach is presented in [9].

One of the first approaches for clustering search results is Scatter/Gather [32, 49]. Scatter/Gather is a document browsing technique that uses document clustering as primary operation. A good discussion on categories and clusters in search results by Hearst is [48].

Document summarization has been an active area of research since the 1950s. A recent article by Spärck Jones covers the current state of the art [56] but there are many other articles and books that discuss other relevant aspects in more detail. Projects that are close to our work are the sentence selection approach presented by Goldstein *et al.* in [41], snippet generation in an information seeking process presented by McDonald [73] and the user study based on snippet length that is described by Kaisser in [57]. Marchionini and White present a framework for user interfaces that support information seeking with an important emphasis on summaries in [71].

5.9 Summary

Temporal information embedded in documents in the form of temporal expressions provide an import means to further improve current search engines and thus user experience by simply applying an additional step to traditional document ranking approaches. We have shown how such a time-based document clustering can be achieved when temporal expressions in documents are readily available

We have introduced the TCluster algorithm that provides great flexibility and allows users to explore clusters of search result documents that are organized along well-defined timelines, supporting different levels of time granularity. The user can also provide temporal expressions are part of a query, as described in some query scenarios.

We defined an alternative document snippet based on temporal information called TSnippet. The idea of using sentences that contain the most frequent and valuable chronons can be used for constructing useful summaries. The evaluation against two Web search engines showed that our technique doubles in precision for snippets that contain temporal information.

The prototypical implementation shows that the proposed technique can effectively be realized using existing tools and systems. Our evaluation demonstrates the utility of the time-based clustering over existing approaches that cluster documents only based on document timestamps.

Chapter 6

Exploratory Search using Timelines

In the last three chapters we have defined a number of techniques for extracting temporal information and analyzing document collections statically and dynamically. All those mechanisms can be implemented as components and later be used to build, for example, an exploratory search system.

In this chapter, we introduce Pacha, an exploratory search system that leverages all the techniques that we have described in earlier chapters. Pacha uses timelines to present and explore search results augmented with named-entities.

This chapter is organized as follows. The next section describes motivation and objectives. Section 6.2 presents an overview of existing and new visualizations for timelines and time-related information. Section 6.3 presents exploratory search in the context of temporal information. The exploratory search system is covered in Section 6.4. Section 6.5 presents the prototypes implemented. The evaluation framework and user study are covered in Section 6.6. Finally, Section 6.7 presents related work.

6.1 Motivation and Objectives

Exploratory search systems have emerged as a specialization of information exploration to support serendipity, learning, and investigation of large data collections. In search situations where the task requires browsing and exploring search results, we argue that temporal information and other named-entities embedded in the data can help accomplish such tasks more effectively.

Users using exploratory search systems require support that extends beyond the known item retrieval that is well handled by many modern search systems. They may require assistance in discovering new associations, navigating complex information spaces, or developing an understanding of terminology. An important part of this research is the design of exploratory search interfaces that support human-information interaction [111, 112].

Current interfaces to search engines typically present search results sorted by the relevance of documents from a document collection (or the Web) to a search query. For this, the freshness of the information, that is, documents or parts thereof, is considered an important part of the result quality. Temporal attributes in Web pages or documents such as date, however, are just viewed as some structured criteria to sort the results in descending order of relevance.

In search situations where the task requires the browsing and exploration of a search result, we argue that temporal information can help significantly to accomplish respective tasks. The presentation of relevant information along a well-defined and understood timeline is an important step to find, for example, the most recent document relevant to a query or the first point in time a document (based on the temporal information contained in the document) is relevant to the query.

To summarize, the objectives of this chapter are to:

(i) Develop a prototype of an exploratory search application that includes timeline visualizations metaphors.

(ii) Precisely define exploratory search in the context of our work.

(iii) Investigate how to use timelines for exploratory purposes.

(iv) Define an evaluation framework and provide experimental results.

6.2 Timelines Visualizations

In this section we provide a high level overview to some existing techniques for visualizing time-based information. We cover technology that is available in commercial products an in open-source toolkits. We also introduce our own visualization called *TTiles*.

6.2.1 SIMILE Timeline

Timeline is a DHTML-based widget for visualizing time-based events, as part of the SIMILE (Semantic Interoperability of Metadata and Information in unLike Environments) project [100]. A timeline contains one or more *bands*, which can be panned infinitely by dragging with the mouse. A band can be configured to synchronize with another band such that panning one band also scrolls the other.

All data (events and time) is specified in XML. This is a good advantage because it is very easy to automate data generation from a database. Figure 6.1 shows a timeline of articles relevant to the query [McCaw] using the TimeBank data set.

6.2.2 TimeWall

TimeWall is an information visualization SDK (Software Development Kit) that allows a user to see patterns over long horizons while being able to focus in on a particular time of interest [102]. Users can point TimeWall at any data set with a temporal element and view it in a new way, leveraging existing data assets. There is also support for filtering based on any combination of structured information such as numeric, geographic, categoric, or other criteria. The filtering is performed on a separate pane that interacts with the main wall.

TimeWall allows a user to define categories on the vertical axis and customize

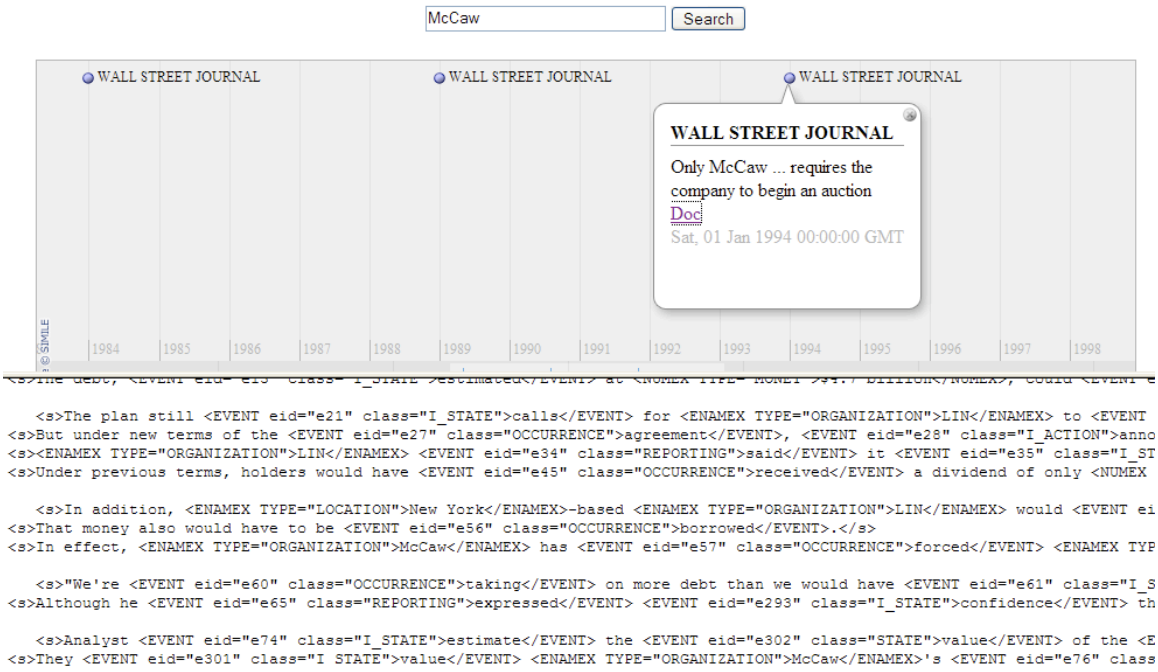


Figure 6.1: TimeBank search results exploration with SIMILE.

the *cards* for specific individual events. There is also a stretch property can be set to dynamically narrow the data range. This is useful for focusing on the last month, the last hour, or any date range.

6.2.3 Sparklines

A sparkline is a type of information graphics that presents in a simple and condensed way trends and variation, associated with a measurement such as average temperature or stock market activity [106]. These are often used as elements of a small multiple with several lines used together.

Figure 6.3 shows an example using the CNN 9/11 news article and Einstein page in Wikipedia. In the case of the Einstein page, the values are normalized to the year chronon. The sparkline is ascending with high peaks. These peaks are references to future chronons in those parts of the document. The CNN 9/11 news article example, in contrast, shows as steady climb in time chronon with a huge dip and then back to where it was before. The explanation relies on a sentence that reference a current event in a time zone and also

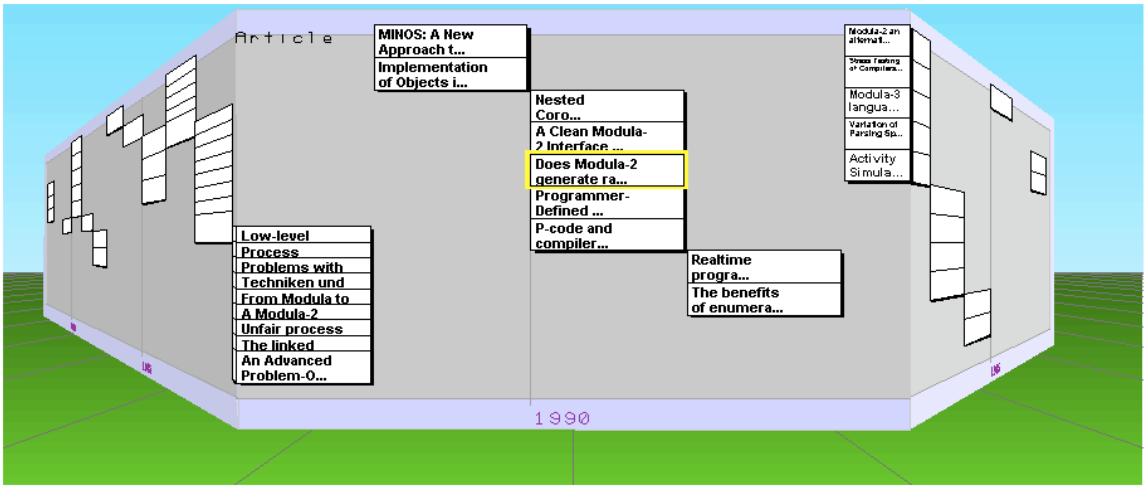


Figure 6.2: DBLP search results exploration with TimeWall.

mentions another event in a complete different time zone, which the temporal tagger is unable to unify. More precisely, the part that references 2:30 a.m. in a sentence that is anchored at 6 p.m.

```
6 p.m.: Explosions are heard in Kabul, Afghanistan ... The attacks
occurred at 2:30 a.m. local time ...
```

This is, of course, not a very common case as one would expect in a typical document. However, as we have seen in previous examples, this shows that NLP can some time produce incorrect answers.

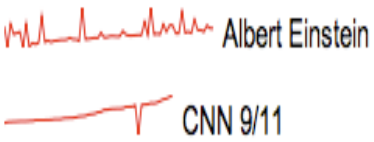


Figure 6.3: Sparklines for two documents.

6.2.4 TTiles

The TTiles (Temporal Tiles) display consists of a rectangular box that contains a text label that defines the document (like a file name or title) and four bars, each repre-

senting 25% percent of the whole document. The construction of a TTile is straightforward and involves looking at the temporal expressions frequencies according their location in each of the four quarters in a document (as introduced in Section 4.4). Each tile is colored according to a value that ranges from the low (blue color) to high (yellow color). The goal is to show in a simple display which part of the document contains most of the temporal expressions. Figure 6.4 shows an example using the CNN 9/11 news article and Einstein page in Wikipedia.

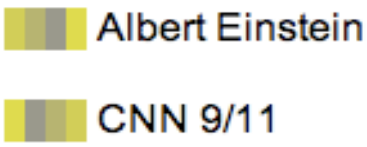


Figure 6.4: The TTile display.

6.3 Exploratory Search

Traditionally, information retrieval systems focus more on algorithms and techniques to further improve the relevance ranking of search results, but keep the presentation of search results very simple. On the other hand, users who engage in information seeking need to identify different cues that might influence the assessment of the relevancy of search results.

As search applications keep gathering new and diverse information sources, a growing number of parameters are used to tune the ranking criteria. At the same time, an increasing set of elements contribute to the detection of *information scents*. Information scents play an important role in guiding users to the information they are looking for, and they also play a role in providing the user with an overall sense of the contents of a document collection.

To address these aspects, our approach involves the (1) identification of certain elements embedded in the content of documents that might be relevant to a user query

and (2) representing such elements as information scents in a user interface for further exploration. Aspects related to having the user determine the relevance of a document or detecting all possible cues in documents are out of the scope of this demonstration. Here we rather concentrate on exploiting temporal information and named-entities, their usage in search and exploration, and enhancing the presentation of search results using techniques that go beyond the traditional top-10 hit list, as realized in most of today's applications.

The presentation of relevant information and search results along a timeline is an important step to find, for example, the most recent document relevant to a query or the first point in time a document is relevant to the query. Other usages are the detection of high activity in a period of time and searching the future. The use of time and timelines for clustering and browsing nicely fits the functionality of exploratory search systems and goes beyond simply returning some documents or answer in response to a user query.

6.4 Pacha Architecture

Our approach supports the exploitation of temporal information in documents, and the usage of such information to anchor search results along a well-defined timeline. We believe that such timelines should be an essential part of every exploratory document search system.

For exploratory search systems, taking advantage of temporal expressions embedded within a document leads to a much richer framework for exploration. We believe this is an important ingredient for the *information forager* who is trying to see the profit in terms of the interaction cost required to gain useful information for an information source [82]. Users tend to prefer sources, in this case search engines, that are richer in good results. These good results involve adding important nuggets, like time.

We now describe the main components of Pacha. The system realizes two main components: 1) a back-end, which supports corpora processing and storage as well as index creation, and 2) a processing unit, which realizes query processing and time-line construction and also implements a user interface to the system. Corpora processing involves using the

temporal document pipeline as described in Section 3.9.

For query processing, Pacha takes some user query terms as input, executes the query, performs document ranking, and generates the appropriate timeline in an intermediate format. The search result is then presented in a particular information visualization metaphor as part of the Web-based user interface.

6.4.1 Operations

For exploring the search results along the timeline, we have identified the following basic operations a user can employ in the discovery process. These operations are presented in the user interface.

(i) First and last. These two operations allow the user to select a document based on the earliest and most recent chronon. For example, the most recent article on a particular topic. The graphical representation of the timeline makes it easier to spot the document.

(ii) Duration. Given a granule as input, the period operator returns all documents that fall into that interval. As example, documents about World War II. We can think of this as a time filter on the search results.

(iii) Activity. This operator takes as input the entire collection and produces the *temporal activity* of a particular topic. The idea behind this is to detect bursts of activity in time by leveraging the temporal information in documents. This is a very practical way to produce a summary of the timeline.

6.4.2 Intermediate Representation

From the information visualization perspective, data flows from its raw form through a number of intermediate steps like *data tables*, *visual structures*, and finally *views*, which are then presented to the user. Traditionally, the most important step is considered to be from data tables to visual structures. We can achieve the separation of concerns

(database from visualization) by framing the model using the standard three-schema architecture (external view, logical view, physical view).

Pacha contains a package that takes an XML representation of the data to be visualized and generates data *exploration views* as the main interface for implementing the separation of concerns. The advantage of data exploration views is that they are always the same while the data sources and visualization metaphors can change.

The data exploration views consist of a number of views for data sources, analytics, and visualizations. The view (`pacha_sources`) contains all data about existing sources in the system. The `pacha_visualization` view represent the visualization metaphors available. The `pacha_exploration` view shows the data exploration techniques that available in the system. Data exploration views represent data that define data sources DS, analytics AN, and visualizations VS.

A *data source* is a collection $\langle D, A \rangle$, where D is a finite set of documents of type `text` and format *f*, and A is a finite set of attributes, each describing the documents. An *analytic operation* AN is a pair $AN = \langle DS_A, A \rangle$ and defines what operation on some attributes A of a data set DS_A can be applied to produce new values. In our case the operation is the timeline generation. Finally, an *information visualization* VS as a pair $VS = \langle DS_A, V \rangle$, which describes what information visualization metaphor on attributes A of a data set DS_A can be applied to produce a visual representation.

6.5 Experimental Prototypes

We built a couple of experimental prototypes using the Pacha architecture and system. The first targets the bibliographic domain where the scenario in mind is to explore past and future references to articles. In the second one, we explore a more traditional search scenario using a Wikipedia data set.

6.5.1 DBLP

The document collection consists of approximately 297,000 journal papers records from the DBLP bibliography data set. In the DBLP collection, a record only contains the bibliographic data like author, title, journal, etc. and not the whole paper. Since the data is available in XML format, we loaded the content into an Oracle database for storage. The extensions to SQL allow us to use XPath and search features for querying the text indexes.

For the visualization part, we use the SIMILE Timeline toolkit [98], for presenting time based events. The exploratory search prototype is a Web-based application and no specific plug-in is required.

The interface is organized as follows. The main section takes half of the screen and contains the search box and the timeline. The timeline consists of two bands that represent different time scales: decade and year. Both bands are synchronized such that panning one band also scrolls the other. The lower band (decade) is much smaller since the goal is to show activity in a decade. The upper band shows all articles in a given year. When the user clicks on an item bullet, the bibliographical information is presented (e.g., author, title, journal, etc.). If the user clicks on the "EE" link (electronic copy), the article content (typically a PDF file) is presented in a separate frame.

Figure 6.5 shows the exploratory search interface in action for the query results about [compiler]. The system retrieves all journal articles that contain "compiler" in the title and returns a hit list clustered by year. All the search results are anchored in the timeline. If more than one article falls within a year, the order is based on its relevance to the query. In this example, we can use the lower band to pick a decade. We can move to the 50s where the early papers on compilers where published or stay in the current year to see the latest publications. Say we are interested in papers from the 60s, because they were very influential. After selecting that particular decade, we can see an interesting number of articles including the one selected about GIER ALGOL.

In the second visualization, we use Inxight's TimeWall [102] to define cards with attributes that allow their placement on the wall along the timeline. In both metaphors

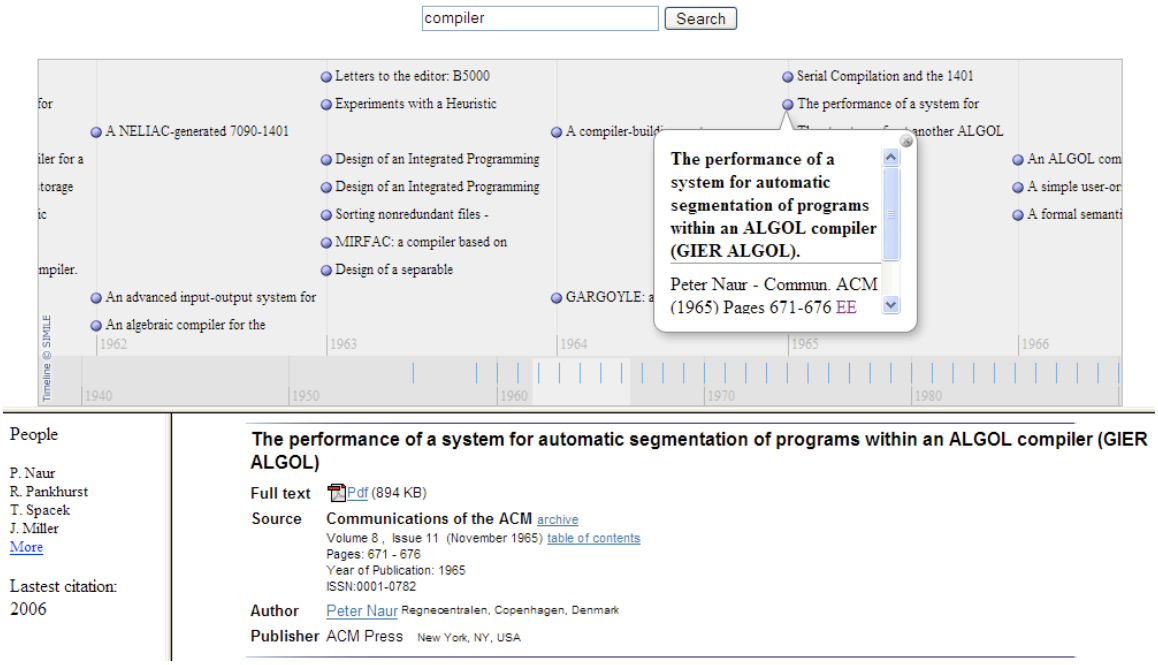


Figure 6.5: Exploration of DBLP search results using Timeline (SIMILE).

one can click a particular document and see the content in a separate frame.

In summary, the user can see all search results in a timeline, observe years of high activity (i.e., many document are relevant to the query terms in respective years), explore the items using both bands and then click to select a particular article. A separate frame shows the complete article as well as other information such as related people and the latest year of a citation.

This output of the search results is an intermediate representation, which is transformed to the timeline visualization format. This is useful in case one wants to use other timeline visualizations. The toolkit is flexible enough so it is also possible to show the same information in a vertical timeline.

Current scholarly literature search applications such as Google Scholar or the ACM Digital Library rank articles by relevance and provide a few links for further exploration, like citations and references, to name a few. When an author is engaged in researching a topic, tracing references back and forth while keeping the original document in focus is a time consuming task when using those systems.

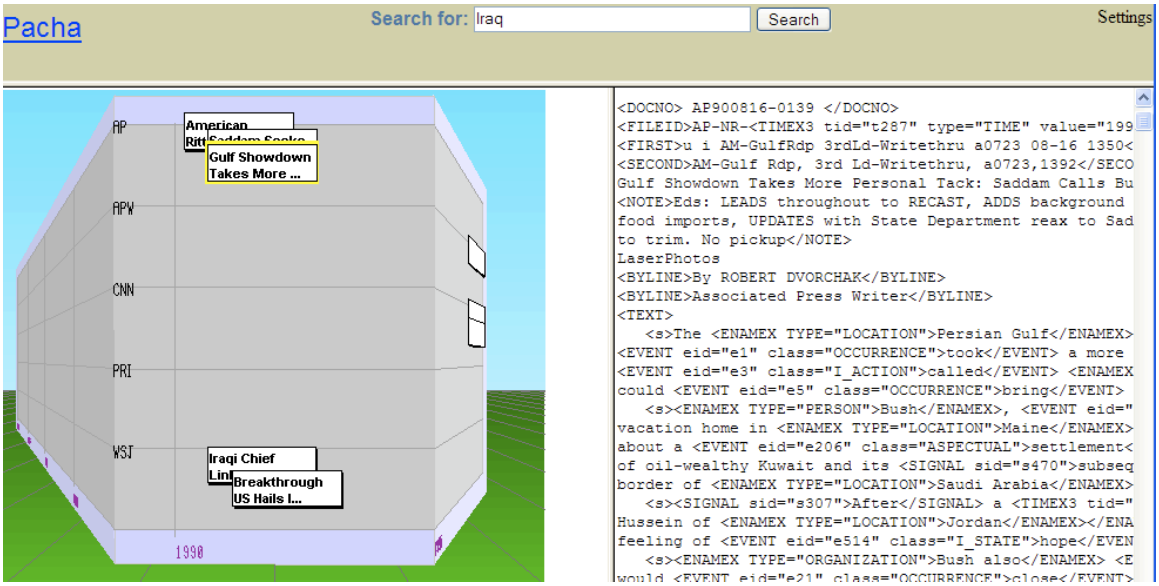


Figure 6.6: Exploration of TimeBank search results using TimeWall.

As another example, say that we would like to replicate the SIGMOD Test of Time for a certain paper. The ACM SIGMOD Test of Time recognizes the best paper from the SIGMOD proceedings 10 years prior (i.e., for 1998 the 1988 proceedings will be consulted), based on the criterion of identifying the paper that has had the most impact (research, products, methodology) over the intervening decade.

With Pacha, we enter a query like [online aggregation] and articles are returned as search result that contains those keywords in the title, sorted by relevance according to the rank function introduced in Section 5.2.2. Once the user selects an article of interest, e.g., J. Hellerstein's SIGMOD 1997 article, the system places it on a timeline (1997). On the same timeline, we can see articles prior to 1997 (references) and after that year in the form of citations. For exploration purposes, the user can see all search results in a timeline, observe years of high activity, use other entities like conference name as cues, and then click to select a particular article to explore further. Figure 6.7 shows the interface for searching and navigating the search results.

All these browsing and exploration tasks are realized using the same single interface and window, ensuring that a user does not loose her focus while searching and exploring

articles. Of course, the above functionality assumes that citations of articles pre-computing, otherwise such a data exploration feature would not be possible.

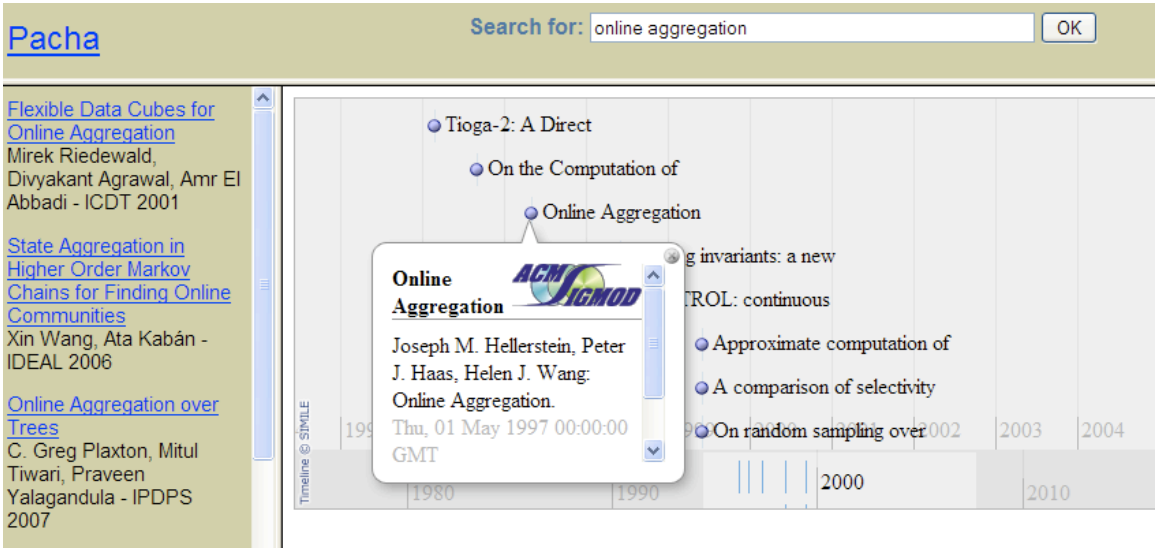


Figure 6.7: Exploring a particular article in the DBLP data set.

Compared to other literature search systems our prototype system allows the user to see the presence of articles in time as well as years of high activity. We believe this type of representation of high activity is crucial for domains where there is a concentration on the research or analysis part

6.5.2 Wikipedia

The second document collection is the featured articles from Wikipedia that were introduced in Chapter 4. Similarly to the previous example, the interface is organized as follows. The left frame contains the search results, in this case augmented by entities (person) and the right frame the timeline. The mechanics of the bands is the same and at the bottom, there are labels for some other entities (location and organization).

Say that we are interested in exploring more about prominent physicists and we enter the query [nobel prize physics]. The engine returns all persons that have the text in the page. Note that this not necessary mean that all results are Nobel Prize winners. On the timeline, we can see those names anchored to a particular year. The timeline allows the

user to see context, which is important and discover, for example, that Robert Oppenheimer is a very relevant result even though he never won the prize. Figure 6.8 shows the interface.

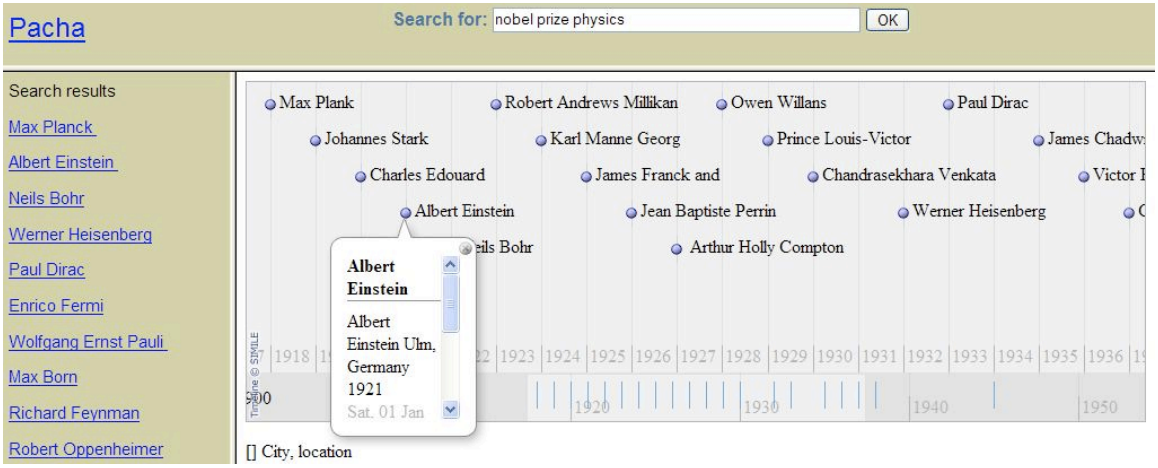


Figure 6.8: Exploring Nobel Prize winners in Wikipedia.

The previous tests with featured articles in Chapter 4 showed that biographies are rich in terms of time information. In this context then, our primary focus is a collection of biographies pages collected from Wikipedia about people (Nobel Prize winners etc.). The user scenario is as follows. Assume one queries the collection for [Einstein], like in a conventional search engine. Our system returns a hit list that contains the usual document information but also includes information about the temporal coverage. Assume the user is specifically interested in the time period 1905–1920. The system determines (based on the histograms constructed for the documents in the hit list) the documents from the above hit list that are temporally rich in this interval and presents the richest ones to the user. It is then possible to refine the above query to get the document that is temporally most specific to that interval (e.g., going from year to days, using the zoom-in operation on histograms, see Section 4.2.2). Finally, the user selects a similar document with respect to the whole Einstein document that shows up first in the hit list.

Like in the other experiments, we use the same temporal annotation pipeline for processing documents. In this particular case, we also built a search engine that returns search results ranked by traditional tf-idf. We augmented the hit-list by providing

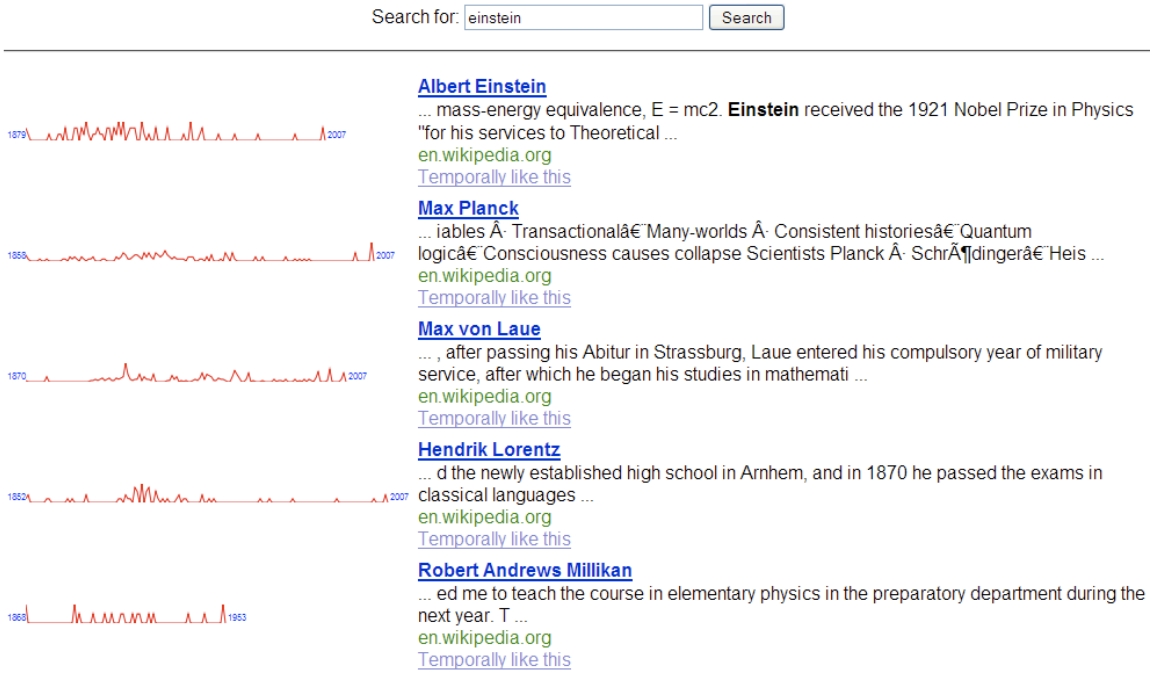


Figure 6.9: Enhanced user interface showing temporal information as sparklines.

a sparkline that displays the temporal coverage of each document. The sparkline was also enlarged with two labels that represent the temporal boundaries t-low and t-high, respectively. Figure 6.9 shows the new hit list with temporal document information to the left of each result document. When the user clicks on the "temporally like this" link, the engine presents another set of documents ordered by their temporal distance to the selected document. As an example, Table 6.1 shows the computed distances based on Einstein's Wikipedia page using the distance formula we presented in Section 4.3.

6.6 Evaluation Framework and Results

One of the main components of Pacha is the user interface that incorporates a novel approach to use timelines for exploration. Designing and implementing usable interfaces involves following establish usability engineering principles like the ones presented by Nielsen in [78]. A complete study from the usability perspective is out of the scope of this research.

There have been only very few studies on the evaluation of time-based exploratory

Distance function value	Web page
6.7338	James Franck
7.1435	William Lawrence Bragg
7.3603	Manne Siegbahn
7.3824	Victor Francis Hess
7.7717	Philipp Lenard
8.1311	Max Planck
8.1618	Owen Willans Richardson
8.4538	Pieter Zeeman
8.7838	Ernest Walton

Table 6.1: Temporal distance to Einstein's Wikipedia page.

search or temporal similarity, with the exceptions of topic tracking [96] and timeline-based user interfaces [12]. We consider our evaluation approach to be closer to the information seeking and retrieval in context as proposed by Ingwersen and Järvelin in [53]. For our experiments, we adopted some of the guidelines that are outlined on Cronopath and Continuum. There are two important features of our approach that we are interested in evaluating: *presentation* and *temporal distance*.

Presentation involves displaying the search results augmented by time information. We provide a display that captures the main temporal features of a Web page, and it is presented along the traditional look and feel. For temporal distance, we add a "temporally like this" function that, given a Web page, returns all pages that are temporally similar to the selected one, ranked by the distance to the selected page (see Section 4.3).

6.6.1 User Survey

We performed a survey among 30 persons (graduate students and faculty) about temporal information to discover alternative exploration scenarios that can be used in future system development. The following is a condensed version of the questions in the survey.

1. Say you issue a query on a Web search engine (like Google, Yahoo!, etc.) about Albert Einstein restricted to the English version of Wikipedia.org. For example the query [albert einstein site:en.wikipedia.org]. Does the engine return temporal information

about those documents in every item of the list? By temporal information we mean anything that references time (e.g., a timestamp or a series of tokens like "October 2002" or "Thanksgiving").

2. Do you think a search engine should present temporal information about documents/Web pages excluding time-stamp as part of search results?

3. In which search/discovery/exploratory scenarios do you think the presentation and usage of temporal information would be useful?

4. In which search/discovery/exploratory scenarios do you think the presentation and usage of temporal information would not be useful?

5. Do you think a snippet covers adequate temporal information about the document/Web page? A snippet is what you see between the title and URL in Web search engine result set. It usually contains a KWIC-like text (or concordance) with some highlighted terms.

6. In which cases does a snippet cover adequate temporal information about the document/Web page?

7. In which cases does a snippet not cover adequate temporal information about the document/Web page?

Table 6.2 shows a summary for questions 1, 2, and 5 respectively. For those questions that required a more elaborate answer beyond a simple yes/no, we have summarized the feedback in tables 6.3, 6.4, 6.5, 6.6, 6.7, and 6.8. In some cases, we did correct grammar or typing errors for clarification purposes.

In some cases the respondents added more feedback in the answers. We now analyze each of the questions in more detail.

Question 1 shows a 70% for the "No" option and 30% for "Do not know". The high percentage is expected due to the fact that most search engines do not leverage temporal

Survey question	Yes	No	Do Not Know
1	0%	70%	30%
2	65%	10%	25%
5	30%	35%	35%

Table 6.2: Answers collected for questions 1, 2, and 5.

information. The surprising factor was on the "Do not know", which we did not expect to exceed 10%. One possible answer is that users did not understand the concept of temporal information well.

For question 2, a high percentage answered positively (65%), the "Do not know" option was in second place (25%) and "No" in third (10%). Table 6.3 provides more additional comments regarding question 2.

Additional comments from question 2
"Maybe. Not on the top of the list of features I want."
"Sometimes dates are helpful as they mark a change, for example tourist info for New Orleans after Katrina might be different. Or events with a difference in perception before/after a phase shifting event."
"Yes, especially in the case where the relevancy of the results to my search is more affected by time (recency) than other metrics."
"If the data is presented in a useful way."

Table 6.3: User survey: additional comments from question 2.

For question 3, the answers can be classified into two main categories: exploration/discovery (*looking for the evolution of the computer industry*, *movie star within a specific time frame*) and question-answering (*when did Albert Einstein start working on physics?*, *winner world soccer championship*). Table 6.4 provides more additional comments regarding question 3.

In question 4, the respondents were unanimous that for certain *timeless* queries, there is no need to add temporal information (*definition of 'color'*, *mathematical facts*, *vegetable soufflé recipe*). Table 6.5 provides more additional comments regarding question 4.

Question 5 shows a tie among the "No" and "Do not know" options at 35%. There was little feedback for this question. Table 6.6 shows a couple of answers.

Additional comments from question 3
"Technology queries, since technology changes so fast."
"News/Current events, geopolitics, science (historical trends eg climate change)."
"When I search a paper, if I cannot remember the specific title or the name of the author, but I can remember the year when it was published, then I can just give some keywords in the title and time for search."
"If I want to find information about a reoccurring event (such as the World Series), but want a specific edition (such as the 2002 World Series)".
"News, current knowledge."
"In most actually. If I'm looking for a piece of information, I want to ensure that my search results are relevant: often times I find web pages from ten years ago when I want information about a product or technology that came out last year. Information about the time that the content is referring to would be useful for more general research purposes; for instance, if I'm looking for information about my grandfather, I only want pages about people, places, and events prior to his death. Likewise, if I'm doing academic or patent research, I might want to find the first paper that discussed a certain idea, or papers at least published before the "seminal" paper on the topic."
"Discovery - seeing older and newer related documents to the search results that I choose to click through on. Search - allow filtering by ranges (newer, older), similar to how Google Scholar does this. Exploratory - faceted browsing (where the facets are in the temporal dimension) would be useful. For example, see markmail.org's useful interface for searching across mailing lists."
"Research of scientific papers, news items, stock quotes, etc."
"If there are several copies (with some different details) of a same document, I'd like to know which one is the origin. If I want to know recent news about something, I don't want the old news about this thing."
"Queries about news articles, financial information, science (e.g., full version of an article out? Follow-up research?) - lots on freshness updates (When I'm looking for news about a piece of software, I want to see which pages have been updated most recently.)"

Table 6.4: User survey: additional comments from question 3.

Additional comments from question 4
"Historical searches, for example - there may be different slants on the data depending on when it was written, but the basic facts don't change much"
"Tutorials and other 'how to' info."
"Slow/never changing data sources like static versions of documents, dictionaries, etc."
Historic documents, articles without an event that changes perception before/after event. Older events, for eg. if I was searching for Fermat's last theorem in '95 I would like to know when the results were published (before the results by Wiles or after). But now the topic has established. I don't care about the dates."
"Old/mature knowledge."
"When the underlying information has less temporal aspects to it (ex: what are the songs lyrics for John Denver's songs?). Or, when the variance over time is low (example: searching for information on a historical event where the knowledge about that event has been relatively static for a given chunk of time)."
"Cooking recipe, etc."
"General info such as definitions."
"If I'm looking for info on well established stuff. Like I want to learn about Albert Einstein for instance. I could care less if it's 1 year old or 50 years old. The page is up to date enough."
"Honestly, I can't really think of any."
"Address lookup."
"Extra information won't hurt."

Table 6.5: User survey: additional comments from question 4.

Additional comments from question 5
"I rarely find snippets to be of any use."
"It varies from snippets to snippets."

Table 6.6: User survey: additional comments from question 5.

For question 6, 30% answered "Do not know", which indicates (similarly to question 5) that the user may not be familiar with the mechanisms behind snippet construction. On the other hand, users who have more expertise and were able to devise how snippets are presented, gave a more detailed explanation as shown in table 6.7.

Additional comments from question 6
"If my search has very specific time like June, 13th, or 2002, then it will be enough."
"When the document/web page can be clearly identified as being relevant/related to a given time, and when the neighboring documents/web pages also have similar time metadata."
"Only when temporal keywords are used during search."
"Actually it depends on Google's snippet extraction algorithm and the location of your search queries. If there is temporal information close to the query you issued, the snippet will probably show it."

Table 6.7: User survey: additional comments from question 6.

Finally, for question 7, 20% answered "Do not know" and 20% "yes" (*Almost always*, *Most of the time*). Table 6.8 presents more comments on this question.

To summarize the findings from the survey, it is clear that time as an alternative dimension for search and exploration would be useful in many scenarios. Snippets, as document surrogates, are a good indication of relevance and therefore should include temporal information. On the other hand, a number of respondents do not see the distinction between "temporal information" and "timestamp". The notion of time data in a document that is not necessarily a creation time seems to be somewhat difficult to understand.

6.6.2 User Study

The final experiments we conducted include identifying particular types of documents and realizing scenarios where our (collection-based) temporal analysis framework can be leveraged for search and document exploration.

As a small-scale user study, we also asked graduate students to perform a task like searching for a person (Einstein in this case), answer the following questions, and provide feedback on the prototype. The main questions included:

Additional comments from question 7
"If I am searching some information during a period like from Jan to Feb, then I guess it will be better for you to cluster the information in some smaller range like per week together."
"Even newspaper articles have the current date (along with the date published somewhere hidden) so finding dates is hard in general (except in blog posts on blogspot which has the URL with info)."
"When temporal information is either inferred from another source; for instance, a page about Albert Einstein might not contain his DOB and DOD, but another, less relevant, page might so the more relevant page could be returned, annotated that the content covers his life from DOB to DOD. Additionally, a snippet doesn't (usually) include information about when the page was last updated, which the search engine could annotate based on the HTTP header for the page."
"When the content does not have a clear time instance or period that apply to all items within the content."
"Very often; but the inaccuracy of the snippets seems to be a more annoying problem."
"In all cases since I don't see temporal info in the snippets."

Table 6.8: User survey: additional comments from question 7

1. Say you issue a query on our prototype about Albert Einstein. Does the engine return temporal information about those documents in every item of the list?

2. Does the combination of sparklines and snippet cover adequate temporal information about the document/Web page?

For question 1, all of them agreed that our approach provided a temporally augmented hit list. With respect to question 2, 85% responded that the combination of sparklines and snippet was a good indication. The overall answer to our prototype was very positive, given that we only provided a very short description with the intent to let the user discover the potential benefits of presenting temporal information.

The overall answer to our prototype was very positive, given that we only provided a very short description with the intent to let them discover the potential benefits of presenting temporal information (see table 6.1).

The evaluation shows that there is evidence that for certain tasks, hit lists re-

turned by current search engines do not provide enough temporal information. Our users were in agreement that the proposed search result list with sparklines shows relevant time information about pages including the similarity feature. There were some disagreements on the combination of snippet and sparklines. For some users, the combination makes sense as long as there is a perceived relationship between the highlighted terms and the visual representation on the left. For others, both are useful and they do not necessarily have to be similar.

This set of experiments was aimed at showing how valuable temporal information can be. The results are encouraging and the user feedback has been very valuable in terms of alternative scenarios for exploration and discovery. There is also a clear need to leverage temporal information in question-answering but this is was not part of the original approach presented in this thesis.

6.7 Related Work

Research in using time for retrieval and browsing activities is fairly recent. The use of tags to visualize photos taken over a period of time is a good example of how useful time can be for arranging objects [36]. Furthermore, in addition to topic detection and tracking, the discovery of bursts in a stream of content can be useful for the identification of topics [59]. In particular, in settings where a user is looking for relevant documents in a less familiar domain, we would like to show peaks of activity (in the form of documents) over time that contain information relevant to the user query.

We agree with previous work that placing search results in a timeline can facilitate the exploration of information [7, 62, 87]. Our approach, however, differs in a number of ways. We do not restrict the search space to a personal desktop environment, because we believe that timelines should be an integral part of search applications. Ringel *et al.* [87] concentrate on "object" timestamp (e.g., a document, email message, or presentation), whereas we use both temporal expressions and document metadata as document timestamps alone can often be misleading. Some experimental ideas by Google for presenting results

in a timeline are restricted to the identification of year only [99]. Exploiting the temporal expressions embedded within a document leads to a much richer framework for search result exploration, which is a core aspect of our work.

There has been a lot of research on document representation and visualization. A similar work is TileBars, a display that presents term distribution and document length information [46]. A core part of TileBars is a technique for segmenting text called TextTiling [47]. The idea of sparklines was designed by Tufte and presented in his excellent book in [106]. Interfaces for photo browsing collections using time is presented by Graham *et al.* in [42]. Other timeline related visualizations are presented by Dubinko *et al.*, Card *et al.*, and Swan and Allan in [36], [22], and [96] respectively.

An earlier version of the exploratory search concepts using timelines is presented in [8]. A demo version using research and commercial visualizations is described in [11].

6.8 Summary

In this chapter we combined all the techniques that we have introduced in earlier chapters and implemented them as components. The goal was to design and implement Pacha, an example of an exploratory search system that uses temporal information for presenting search results along a timeline and exploring documents based on time.

We adapted Pacha and produced a couple of prototypes for two different domain verticals: bibliographic material based on DBLP and Wikipedia using featured articles and Nobel Prize winners biographies.

Pacha's architecture allows the integration of different visualization components as the examples that we have presented. As an example of an alternative visualization, we introduced a display called TTitles for representing the distribution of temporal expressions in a document.

We conducted a user survey about temporal information in current search applications that provided useful information in the implementation stage of the prototypes. We also conducted a user study to gather feedback from our exploratory search designs.

Overall, the findings have been encouraging.

Adding time as part of exploratory search tasks can lead to interesting discoveries. We propose to use well-defined timelines as an alternative view for presenting search results that have rich temporal expressions or where the usage of time plays an important role.

Chapter 7

Conclusions and Outlook

Temporal information embedded in documents in the form of temporal expressions offer an interesting means to further enhance the functionality of current information retrieval and document search applications. Recognizing such temporal information and utilizing it for the exploration of document collections can significantly improve the current functionality of search applications.

As information retrieval applications gather and archive massive data sets, accessing and presenting information that has temporal relevance become more important. We have presented a comprehensive framework that provides the very basis for such functionality.

Using our prototype system, which uses standard software components, we have shown that temporal expressions can be effectively extracted from different document collections. In particular, we introduced novel temporal document measures as basic ingredients for a temporal analysis framework for documents and document collections.

As shown in the context of document collections from Wikipedia, not every collection is sufficiently rich in terms of temporal information. However, as our experiments and usage scenarios demonstrate, the proposed temporal document measures and (collection) histograms not only provide important means to determine such properties for a collection but they also realize novel exploration and search features that can easily be integrated in

standard search interfaces. Augmenting search results with temporal measures helps the user to navigate and explore a temporally rich document collection or hit list.

Our TCluster algorithm provides great flexibility and allows users to explore clusters of search result documents that are organized along well-defined timelines, supporting different levels of time granularity. The evaluation demonstrates the utility of the time-based clustering over existing approaches that cluster documents only based on document timestamps.

We have studied how temporal expressions appear progressively in a document. In the first few paragraphs the reference to time occurs early in the sentence. We use these findings to design a temporal snippet algorithm called TSnippet. The evaluation demonstrated that TSnippet can be an interesting alternative for constructing document previews that highlight time.

The implementation of Pacha, as an example of an exploratory search application, showed that all the proposed techniques can effectively be realized using and extending existing tools and systems. Our evaluation demonstrate the utility of the temporal augmented search result list over existing approaches.

In general, a key aspect of our work is that we believe that when a user is engaged in tasks that require time-related investigations and sense-making, traditional information retrieval and search engines fall short. The use of time information for presenting and browsing nicely fits exploratory search systems that go beyond simply returning some documents or answer in response to a query.

This dissertation explored the usage of temporal information in documents in the context of information retrieval and related applications. The framework we introduced serves as the basis for further temporal exploration and search tasks that are currently being developed. This includes in particular classification methods (topic detection) for temporally rich documents and exploiting the order in which temporal expressions occur in a document, the latter leading to an order-sensitive extension of our temporal coverage technique.

7.1 Future Work

In this subsection we outline a number of extensions to our research.

1. **Similarity measures**. Further work can be done with temporal similarity measures. It would be interesting to combine histograms and t-order for a new similarity measure, as one example. Another alternative would be to incorporate the co-occurrence of the most frequent terms with the most frequent chronons.

2. **Ranking functions**. The ranking functions for TCluster and TSnippet leave much room for further investigations. A possible extension is to use a machine learning approach to automatically derive the values for each parameter, instead of manual experimentation.

3. **Query languages and user interfaces**. Extensions to a query language that is temporally aware would be a desirable feature that gives the user more control for expressing time information. At the same time, one can expose temporal functionally at the user interface layer in many ways, not just timelines but also filters, default values, and similar interaction techniques.

4. **Evaluation**. One aspect that needs further work is to define an adequate evaluation benchmark for temporal-related applications. In the spirit of TREC, it would be useful to provide a separate track that includes particular retrieval tasks in the context of temporal access. At the same time, a set of guidelines for evaluating interaction techniques based on timelines, would also be useful to perform comparisons with other systems.

5. **User studies**. The survey and user study that we conducted showed that there is a wide range of possible scenarios where temporal information can be applied. In our particular case, the population was mainly researchers that were experts in computers and different information seeking strategies. It would interesting to include more

end consumers in a user study with other types of tasks that might not necessarily be classified as of exploratory and/or discovery nature.

With the pervasive usage of other devices to access information online like mobile phones, new user interfaces and visualization metaphors can definitely improve the overall user experience. So far, we have researched advanced representations that are suitable for traditional desktop machines. More work needs to be done to study timelines, including new designs, for such devices.

6. **Classification**. Another application of the framework that we provided, would be to classify documents based on their temporal information. For example, given a corpora, it should be possible to identify which documents are news, résumés, bibliographies, etc. A solution for classifying those documents is to construct a decision tree that incorporates the features tests that are needed to discriminate between documents. All concepts that we have introduced earlier like temporal specificity and temporal coverage, would be part of the decision tree technique.

In summary, exciting and promising research awaits to unlock the value of temporal information in many possible applications.

Appendix A

Example of a Temporal Document Processing Pipeline

In this appendix we show the temporal annotation pipeline introduced in Chapter 3 using as example Einstein's page in Wikipedia. We use the following text fragment to showcase the main steps.

```
Following graduation, Einstein could not find a teaching post. After almost
two years of searching, a former classmate's father helped him get a job
in Bern, at the Federal Office for Intellectual Property, the patent office,
as an assistant examiner. In 1903, Einstein's position at the Swiss Patent
Office was made permanent. Einstein's college friend, Michele Besso, also
worked at the patent office. With friends they met in Bern, they formed a
weekly discussion club on science and philosophy, jokingly named "The
Olympia Academy". Einstein married Mileva Maric on January 6, 1903, and
their relationship was, for a time, a personal and intellectual
partnership.
```

The Alembic POS tagger takes as input the original text and returns a new document marked-up with tags following the Brown/Penn convention as introduced in subsection 2.3.1. In a post-process step we include sentence delimiters `<s>` and `</s>` and add two tags at the beginning and end of document respectively (`<DOC>` and `</DOC>`). We also include a date as anchor for resolving temporal expressions. Because we use content from Wikipedia,

we set the date value to the crawl timestamp. All of these would later be needed for the temporal tagger.

```
<DOC> <date_time> 5 May 2008</date_time> ...

<lex pos="VBG">Following</lex><lex pos="NN">graduation</lex><lex
pos="COMMA">,</lex><lex pos="NNP">Einstein</lex><lex pos="MD">could
</lex><lex pos="RB">not</lex><lex pos="VB">find</lex><lex pos="DT">
a</lex><lex pos="NN">teaching</lex><lex pos="NN">post</lex><lex pos=
"PERIOD">.</lex></s><s><lex pos="IN">After</lex><lex pos="RB">almost
</lex><lex pos="CD">two</lex><lex pos="NNS">years</lex><lex pos="IN">
of</lex><lex pos="VBG">searching</lex><lex pos="COMMA">,</lex><lex
pos="DT">a</lex><lex pos="JJ">former</lex><lex pos="NN">classmate
</lex><lex pos="POS">'s</lex><lex pos="NN">father</lex><lex pos="VBD">
helped</lex><lex pos="PRP">him</lex> <lex pos="VBP">get</lex><lex
pos="DT">a</lex><lex pos="NN">job</lex><lex pos="IN">in</lex><lex
pos="NNP">Bern</lex><lex pos="COMMA">,</lex><lex pos="IN">at</lex>
<lex pos="DT">the</lex><lex pos="NNP">Federal</lex><lexpos="NNP">Office
</lex><lex pos="IN">for</lex><lex pos="NNP">Intellectual</lex><lex
pos="JJ">Property,</lex><lex pos="COMMA">,</lex><lex pos="DT">the</lex>
<lex pos="NN">patent</lex><lex pos="NN">office</lex><lex pos="COMMA">,
</lex><lex pos="IN">as</lex><lex pos="DT">an</lex><lex pos="NN">
assistant</lex><lex pos="NN">examiner</lex><lex pos="PERIOD">.</lex></s>
<s><lex pos="IN">In</lex> <lex pos="CD">1903</lex><lex pos="COMMA">,
</lex> <lex pos="NNP">Einstein</lex><lex pos="POS">'s</lex><lex pos="NN">
position</lex><lexpos="IN">at</lex><lex pos="DT">the</lex><lex pos="NNP">
Swiss</lex><lex pos="NNP">Patent</lex><lex pos="NNP">Office</lex><lex
pos="VBD">was</lex><lex pos="VBN">made</lex><lex pos="JJ">permanent</lex>
<lex pos="PERIOD">.</lex><lex pos="NNP">Einstein</lex><lex pos="POS">'s
</lex><lex pos="NN">college</lex><lex pos="NN">friend</lex><lex pos=
"COMMA">,</lex><lex pos="NNP">Michele</lex><lex pos="NNP">Besso</lex>
<lex pos="COMMA">,</lex><lex pos="RB">also</lex><lex pos="VBD">worked
</lex><lex pos="IN">at</lex><lex pos="DT">the</lex><lex pos="NN">patent
</lex><lex pos="NN">office</lex><lex pos="PERIOD">.</lex></s><s><lex
pos="IN">With</lex><lex pos="NNS">friends</lex><lex pos="PRP">they</lex>
<lex pos="VBD">met</lex><lex pos="IN">in</lex><lex pos="NNP">Bern</lex>
<lex pos="COMMA">,</lex><lex pos="PRP">they</lex><lex pos="VBD">formed
</lex><lex pos="DT">a</lex><lex pos="JJ">weekly</lex><lex pos="NN">
discussion</lex><lex pos="NN">club</lex><lex pos="IN">on</lex><lex
pos="NN">science</lex><lex pos="CC">and</lex><lex pos="NN">philosophy
</lex><lexpos="COMMA">,</lex><lex pos="RB">jokingly</lex><lex pos="VBN">
named</lex><lex pos="QUOTE">"</lex><lex pos="DT">The</lex><lex pos="NNP">
Olympia</lex><lex pos="NNP">Academy</lex><lex pos="QUOTE">"</lex><lex pos=
"PERIOD">.</lex></s><s><lex pos="NNP">Einstein</lex><lex pos="VBD">married
```

```
</lex><lex pos="NNP">Mileva</lex><lex pos="NNP">Maric</lex><lex pos="IN">
on</lex><lex pos="NNP">January</lex><lex pos="CD">6</lex><lex pos="COMMA">
,</lex><lex pos="CD">1903</lex><lex pos="COMMA">,</lex><lex pos="CC">and
</lex><lex pos="PRP">their</lex><lex pos="NN">relationship</lex><lex pos=
"VBD">was</lex><lex pos="COMMA">,</lex><lex pos="IN">for</lex><lex pos=
"DT">a</lex><lex pos="NN">time</lex><lex pos="COMMA">,</lex><lex pos="DT">
a</lex><lex pos="JJ">personal</lex><lex pos="CC">and</lex><lex pos="JJ">
intellectual</lex><lex pos="NN">partnership</lex><lex pos="PERIOD">.
</lex></s><s>
</DOC>
```

Finally, using the POS version of a document as input, the GUTime tagger then produces a document marked-up in TimeML. For the above POS version, the following document is obtained.

```
Following graduation, Einstein could not find a teaching post. After
almost <TIMEX3 tid="t23" TYPE="DURATION" VAL="P2Y">two years</TIMEX3>
of searching, a former classmate's father helped him get a job in Bern,
at the Federal Office for Intellectual Property, the patent office, as
an assistant examiner.
In <TIMEX3 tid="t24" TYPE="DATE" VAL="1903">1903</TIMEX3>, Einstein's
position at the Swiss Patent Office was made permanent. Einstein's
college friend, Michele Besso, also worked at the patent office. With
friends they met in Bern, they formed a <TIMEX3 tid="t25" TYPE="DATE"
PERIODICITY="F1W">weekly </TIMEX3> discussion club on science and
philosophy, jokingly named "The Olympia Academy". Einstein married
Mileva Maric on <TIMEX3 tid="t26" TYPE="DATE" VAL="19030106">January 6,
1903</TIMEX3>, and their relationship was, for a time, a personal and
intellectual partnership.
```

Bibliography

[1] Witold Abramowicz and Jakub Piskorski. *Information Extraction From Free-text Business Documents*, pages 12–23. 2003.

[2] Reema Al-Kamha and David W. Embley. Grouping Search-engine Returned Citations for Person-name Queries. In *WIDM '04: Proceedings of the 6th Annual ACM International Workshop on Web Information and Data management*, pages 96–103, 2004.

[3] Alembic. http://www.mitre.org/tech/alembic-workbench/).

[4] James Allan, Rahul Gupta, and Vikas Khandelwal. Temporal Summaries of News Topics. In *SIGIR '01: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 10–18, 2001.

[5] James Allan, Ron Papka, and Victor Lavrenko. On-line New Event Detection and Tracking. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 37–45, 1998.

[6] James F. Allen. Maintaining Knowledge About Temporal Intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[7] Robert B. Allen. A Focus-context Browser for Multiple Timelines. In *JCDL '05: Proceedings of the 5th ACM/IEEE-CS joint Conference on Digital libraries*, pages 260–261, 2005.

[8] Omar Alonso, Ricardo Baeza-Yates, and Michael Gertz. Exploratory Search using Timelines. In Ryen White, Steven Drucker, Gary Marchionini, Marti Hearst, and m.c. schraefel, editors, *Exploratory Search and HCI Workshop*. CHI, 2007.

[9] Omar Alonso and Michael Gertz. Clustering of Search Results Using Temporal Attributes. In *SIGIR '06: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 597–598, 2006.

[10] Omar Alonso, Michael Gertz, and Ricardo Baeza-Yates. On the Value of Temporal Information in Information Retrieval. *SIGIR Forum*, 41(2):35–41, 2007.

[11] Omar Alonso, Michael Gertz, and Ricardo Baeza-Yates. Search Results Using Timeline visualizations. In *SIGIR '07: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 908, 2007.

[12] Paul André, Max L. Wilson, Alistair Russell, Daniel A. Smith, Alisdair Owens, and m.c. schraefel. Continuum: Designing Timelines for Hierarchies, Relationships and Scale. In *UIST '07: Proceedings of the 20th Annual ACM symposium on User interface software and technology*, pages 101–110, 2007.

[13] Internet Archive. (http://www.archive.org).

[14] Featured Articles. (http://en.wikipedia.org/wiki/wikipedia:featured_articles).

[15] Anne Aula, Natalie Jhaveri, and Mika Käki. Information Search and Re-access Strategies of Experienced Web Users. In *WWW '05: Proceedings of the 14th International Conference on World Wide Web*, pages 583–592, 2005.

[16] Ricardo Baeza-Yates. Searching the Future. In *SIGIR Workshop MF/IR*, 2005.

[17] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.

[18] Richard Belew. *Finding Out About*. Cambridge University Press, 2000.

[19] Pia Borlund. The Concept of Relevance in IR. *Journal of the American Society for Information Science and Technology*, 54:913–925, 2003.

[20] Philip Bramsen, Pawan Deshpande, Yoong Keok Lee, and Regina Barzilay. Finding Temporal Order in Discharge Summaries. In *Proceedings of AMIA'06, American Medical Informatics Association*, 2006.

[21] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers Inc., 1999.

[22] Stuart K. Card, Bongwon Suh, Bryan Pendleton, Jeffrey Heer, and John W. Bodnar. Timetree: Exploring Time Changing Hierarchies. In *IEEE Symposium on Visual Analytics Science and Technology (VAST)*, 2006.

[23] Roberta Catizone, Angelo Dalli, and Yorick Wilks. Evaluating Automatically Generated Timelines From the Web. In *5th International Conference on Language Resources and Evaluation, Genoa, Italy*, 2006.

[24] Sung-Hyuk Cha and Sargur N. Srihari. On Measuring the Distance Between Histograms. *Pattern Recognition*, 35(6):1355–1370, June 2002.

[25] Soumen Chakrabarti. *Mining the Web*. Morgan Kaufmann, 2002.

[26] Hai Leong Chieu, Hwee Tou Ng, and Yoong Keok Lee. Closing the Gap: Learning-based Information Extraction Rivaling Knowledge-engineering Methods. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 216–223, 2003.

[27] Lois C. Childs and David Cassel. Extracting and Normalizing Temporal Expressions. In *Proceedings of ACL Workshop*, pages 51–56. Association for Computational Linguistics, 1996.

[28] Jennifer Chu-Carroll, John Prager, Krzysztof Czuba, David Ferrucci, and Pablo Duboue. Semantic Search via XML Fragments: A High-precision Approach to IR. In *SIGIR '06: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 445–452, 2006.

[29] Jim Cowie and Wendy Lehnert. Information Extraction. *Commun. ACM*, 39(1):80–91, 1996.

[30] Fabio Crestani, Mounia Lalmas, Cornelis J. Van Rijsbergen, and Iain Campbell. Is this Document Relevant? Probably: A Survey of Probabilistic Models in Information Retrieval. *ACM Comput. Surv.*, 30(4):528–552, 1998.

[31] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *ACL '02: Association for Computational Linguistics*, 2002.

[32] Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey. Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections. In *SIGIR '92: Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 318–329, 1992.

[33] DBLP. (http://dblp.uni-trier.de/).

[34] Fernando Diaz and Rosie Jones. Using Temporal Profiles of Queries for Precision Prediction. In *SIGIR '04: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 18–24, 2004.

[35] DMOZ. (http://www.dmoz.org).

[36] Micah Dubinko, Ravi Kumar, Joseph Magnani, Jasmine Novak, Prabhakar Raghavan, and Andrew Tomkins. Visualizing Tags over Time. In *WWW '06: Proceedings of the 15th International Conference on World Wide Web*, pages 193–202, 2006.

[37] Albert Einstein. (http://en.wikipedia.org/wiki/einstein/).

[38] 1999 Information Extraction Entity Recognition Evaluation. (http://www.itl.nist.gov/iad/894.01/tests/ie-er/er_99/er_99.htm).

[39] Paolo Ferragina and Antonio Gulli. A Personalized Search Engine based on Web-snippet Hierarchical Clustering. In *WWW '05: Special interest tracks and posters of the 14th International Conference on World Wide Web*, pages 801–810, 2005.

[40] Petr Gardenfors. *Conceptual Spaces*. The MIT Press, 2004.

[41] Jade Goldstein, Mark Kantrowitz, Vibhu Mittal, and Jaime Carbonell. Summarizing Text Documents: Sentence Selection and Evaluation Metrics. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 121–128, 1999.

[42] Adrian Graham, Héctor García-Molina, Andreas Paepcke, and Terry Winograd. Time as Essence for Photo Browsing Through Personal Digital Libraries. In *JCDL '02: Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital libraries*, pages 326–335, 2002.

[43] Jens Graupmann, Ralf Schenkel, and Gerhard Weikum. The Sphere Search Engine for Unified Ranked Retrieval of Heterogeneous XML and Web Documents. In *VLDB '05: Proceedings of the 31st International Conference on Very Large Data Bases*, pages 529–540. VLDB Endowment, 2005.

[44] David Grossman and Ophir Frieder. *Information Retrieval*. Springer, 2004.

[45] GUTime. (http://complingone.georgetown.edu/ linguist/).

[46] Marti A. Hearst. Tilebars: Visualization of Term Distribution Information in Full Text Information Access. In *Proceedings of the SIGCHI Conference on Human factors in Computing Systems CHI '95*, 1995.

[47] Marti A. Hearst. TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages. *Computational Linguistics*, 23, 1997.

[48] Marti A. Hearst. The Use of Categories and Clusters for Organizing Retrieval Results. In *Natural Language Information Retrieval*, 1999.

[49] Marti A. Hearst, David R. Karger, and Jan O. Pedersen. Scatter/Gather as a Tool for the Navigation of Retrieval Results. In *Working Notes of the 1995 AAAI Fall Symposium on AI Applications in Knowledge Navigation and Retrieval*, 1995.

[50] Lynette Hirschman. Retrieving Time Information from Natural-Language Texts. In *SIGIR '80: Proceedings of the 3rd Annual ACM Conference on Research and Development in Information Retrieval*, pages 154–171, 1981.

[51] Jerry Hobbs and Feng Pan. An Ontology of Time for the Semantic Web. *ACM Transactions on Asian Language Information Processing*, 3(1):66–85, March 2004.

[52] George Hripcsak, Li Zhou, Simon Parsons, Amar Das, and Stephen Johnson. Modeling Electronic Discharge Summaries as a Simple Temporal Constraint Satisfaction Problem. *Journal of the American Medical Informatics Association*, 12(12):55–63, January/February 2005.

[53] Peter Ingwersen and Kalervo Järvelin. *The Turn: Integration of Information Seeking and Retrieval in Context*. Springer-Verlag, 2005.

[54] Peter Jackson and Isabelle Moulinier. *Natural Language Processing for Online Applications*. John Benjamins, 2002.

[55] A. K. Jain, M. N. Murty, and P. J. Flynn. Data Clustering: A Review. *ACM Comput. Surv.*, 31(3):264–323, 1999.

[56] Karen Spärck Jones. Automatic Summarising: The State of the Art. *Inf. Process. Manage.*, 43(6):1449–1481, 2007.

[57] Michael Kaisser, Marti A. Hearst, and John B. Lowe. Improving Search Results Quality by Customizing Summary Lengths. In *Proceedings of ACL-08: HLT*, pages 701–709. Association for Computational Linguistics, June 2008.

[58] Pawel Jan Kalczynski and Amy Chou. Temporal Document Retrieval Models for Business News Archives. *Information Processing & Management*, 41:635–650., 2005.

[59] Jon Kleinberg. Bursty and Hierarchical Structure in Streams. In *KDD '02: Proceedings of the eighth ACM SIGKDD International Conference on Knowledge discovery and Data mining*, pages 91–101, 2002.

[60] Mei Kobayashi and Koichi Takeda. Information Retrieval on the Web. *ACM Comput. Surv.*, 32(2):144–173, 2000.

[61] D. Koen and W. Bender. Time frames: Temporal Augmentation of the News. *IBM System Journal*, 39(4):597–616, 2000.

[62] Aparna Krishnan and Steve Jones. Timespace: Activity-based Temporal Visualisation of Personal Information Spaces. *Personal Ubiquitous Comput.*, 9(1):46–65, 2005.

[63] David M. Levy. *Scrolling Forward*. Arcade Publishing, New York, 2001.

[64] LingPipe. (http://www.alias-i.com/lingpipe/).

[65] Juha Makkonen and Helena Ahonen-Myka. Utilizing Temporal Information in Topic Detection and Tracking. In *ECDL'03, 7th European Conference on Digital Libraries.*, pages 393–404, 2003.

[66] Inderjeet Mani, James Pustejovsky, and Robert Gaizauskas. *The Language of Time*. Oxford University Press, 2005.

[67] Inderjeet Mani, James Pustejovsky, and Beth Sundheim. Introduction to the Special Issue on Temporal Information Processing. *ACM Transactions on Asian Language Information Processing*, 3(1):1–10, March 2004.

[68] Inderjeet Mani, George Wilson, Lisa Ferro, and Beth Sundheim. Guidelines for Annotating Temporal Information. In *HLT '01: Proceedings of the first International Conference on Human language technology Research*, pages 1–3, 2001.

[69] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *An Introduction to Information Retrieval*. Cambridge University Press, 2008.

[70] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[71] Gary Marchionini and Ryen White. Find What You Need, Understand What You Find. *International Journal of Human-Computer Interaction*, 23(3):205–237, 2007.

[72] Andrew McCallum. Information Extraction: Distilling Structured Data From Unstructured Text. *ACM Queue*, 3(9):48–57, 2005.

[73] Daniel M. McDonald and Hsinchun Chen. Summary in Context: Searching Versus Browsing. *ACM Trans. Inf. Syst.*, 24(1):111–141, 2006.

[74] Qiaozhu Mei and ChengXiang Zhai. Discovering Evolutionary Theme Patterns From Text: An Exploration of Temporal Text Mining. In *KDD '05: Proceeding of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data mining*, pages 198–207, 2005.

[75] Stefano Mizzaro. Relevance: The Whole History. *J. Am. Soc. Inf. Sci.*, 48(9):810–832, 1997.

[76] Stefano Mizzaro. How many Relevances in Information Retrieval? *Interacting with Computers*, 10(3):303–320, 1998.

[77] Philippe Muller and Xavier Tannier. Annotating and Measuring Temporal Relations in Texts. In *COLING '04: Proceedings of the 20th International Conference on Computational Linguistics*, page 50. Association for Computational Linguistics, 2004.

[78] Jakob Nielsen. *Usability Engineering*. Morgan Kaufmann, 1994.

[79] Sérgio Nunes, Cristina Ribeiro, and Gabriel David. Use of Temporal Expressions in Web Search. In *ECIR'08, 30th European Conference on IR Research*, pages 580–584, 2008.

[80] Gultekin Ozsoyoglu and Ricahrd T. Snodgrass. Temporal and Real-time Databases: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 7:513–532, 1995.

[81] Marius Pasca. Towards Temporal Web Search. In *SAC '08: Proceedings of the 2008 ACM symposium on Applied Computing*, pages 1117–1121, 2008.

[82] Peter Pirolli. *Information Foraging Theory*. Oxford University Press, 2007.

[83] James Pustejovsky. Time and the Semantic Web. In *TIME*, pages 5–8, 2005.

[84] James Pustejovsky, José M. Castaño, Robert Ingria, Roser Sauri, Robert J. Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R. Radev. Timeml: Robust Specification of Event and Temporal Expressions in Text. In *New Directions in Question Answering*, pages 28–34, 2003.

[85] Arun Qamra, Belle Tseng, and Edward Y. Chang. Mining Blog Stories Using Community-based and Temporal Clustering. In *CIKM '06: Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, pages 58–67, 2006.

[86] Dragomir R. Radev and Weiguo Fan. Automatic Summarization of Search Engine Hit Lists. In *Proceedings of the ACL-2000 Workshop on Recent advances in Natural Language Processing and Information Retrieval*, pages 99–109, 2000.

[87] Meredith Ringel, Edward Cutrell, Susan Dumais, and Eric Horvitz. Milestones in Time: The Value of Landmarks in Retrieving Information from Personal Stores. In *Proceedings of Interact 2003*, 2003.

[88] Tefko Saracevic. Relevance: A Review of the Literature and a Framework for Thinking on the Notion in Information Science. Part II: Nature and Manifestations of Relevance. *JASIST*, 58(13):1915–1933, 2007.

[89] Tefko Saracevic. Relevance: A Review of the Literature and a Framework for Thinking on the Notion in Information Science. Part III: Behavior and Effects of Relevance. *JASIST*, 58(13):2126–2144, 2007.

[90] Sarbanes-Oxley. (http://www.sec.gov/spotlight/sarbanes-oxley.htm).

[91] Frank Schilder. Extracting Meaning From Temporal Nouns and Temporal prepositions. *ACM Transactions on Asian Language Information Processing*, 3(1):33–50, March 2004.

[92] Frank Schilder and Christopher Habel. From Temporal Expressions to Temporal Information: Semantic Tagging of News Messages. In *ACL'01 Workshop on Temporal and Spatial Information Processing*, 2001.

[93] Benyah Shaparenko, Rich Caruana, Johannes Gehrke, and Thorsten Joachims. Identifying Temporal Patterns and Key Players in Document Collections. *Proc. IEEE ICDM Workshop on Temporal Data Mining: Algorithms, Theory and Applications (TDM-05)*, pages 165–174, 2005.

[94] Ben Shneiderman. The Eyes Have it: A Task by Data Type Taxonomy for Information Visualizations. In *Proceedings of IEEE Visual Languages*, 1996.

[95] Richard Snodgrass. *The TSQL2 Temporal Query Language*. Kluwer Academic Publishers, 1995.

[96] Russell Swan and James Allan. Automatic Generation of Overview Timelines. In *SIGIR '00: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 49–56, 2000.

[97] Inxight ThingFinder. (http://www.inxight.com/products/sdks/tf/).

[98] TimeLine. (http://simile.mit.edu/timeline/).

[99] Google Timeline. (http://www.google.com/experimental/).

[100] SIMILE Timeline. (http://simile.mit.edu/timeline/).

[101] TimeML. Specification 1.2.1 (http://www.timeml.org).

[102] Inxight TimeWall. (http://www.inxight.com/products/sdks/tw/).

[103] Hiroyuki Toda and Ryoji Kataoka. A Clustering Method for News Articles Retrieval System. In *WWW '05: Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, pages 988–989, 2005.

[104] Hiroyuki Toda and Ryoji Kataoka. A Search Result Clustering Method Using Informatively Named Entities. In *WIDM '05: Proceedings of the 7th Annual ACM International Workshop on Web Information and Data management*, pages 81–86, 2005.

[105] TREC. (http://trec.nist.gov).

[106] Edward Tufte. *Beatiful Evidence*. Graphic Press, 2006.

[107] Johan van Benthem. *The Logic of Time*. Kluwer Academic Publishers, 1991.

[108] C. van Rijsbergen. *Information Retrieval*. Butterworths,, 1979.

[109] Marc Verhagen, Inderjeet Mani, Roser Sauri, Robert Knippen, Seok Bae Jang, Jessica Littman, Anna Rumshisky, John Phillips, and James Pustejovsky. Automating Temporal Annotation with TARSQI. In *ACL '05: Proceedings of the ACL 2005 on Interactive Poster and Demonstration Sessions*, pages 81–84, 2005.

[110] Vivísimo. (http://www.vivisimo.com). Also known as Clusty.com.

[111] Ryen W. White, Bill Kules, Steven M. Drucker, and m.c. schraefel. Introduction. *Commun. ACM*, 49(4):36–39, 2006.

[112] Ryen W. White, Gary Marchionini, and Gheorghe Muresan. Evaluating Exploratory Search Systems: Introduction to Special Topic Issue of Information Processing and Management. *Inf. Process. Manage.*, 44(2):433–436, 2008.

[113] Dominic Widdows. *Geometry and Meaning*. CLSI, 2004.

[114] Wikipedia. (http://www.wikipedia.org/).

[115] Oren Zamir and Oren Etzioni. Web Document Clustering: A Feasibility Demonstration. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 46–54, 1998.