

Extraction of Temporal Information from Documents

Nattiya Kanhabua

Dept. of Computer Science,
Norwegian University of Science and Technology,
Trondheim, Norway

Abstract. Temporal information in the document is a demanding dimension to be discovered. Recently, many works from board research areas such as database, information retrieval, text mining pay attention to a temporal aspect. In this paper, we give a survey of state-of-art in extracting temporal information from document collections. As it is quite a new discipline, there is no standard comparison scheme. Consequently, we have proposed a descriptive framework for an analysis of existing works in this area.

1 Introduction

As the Internet becomes one of most important communication backbones in this decade, many people are able to consume information from the Internet easily. Furthermore, they also can produce information into the Internet. Those information are represented in digital formats with several forms, for example, webpage (newspaper, or personal blog), text file (word processing or PDF), or multimedia (music files or pictures). As a result, the size of documents in the Internet is increased dramatically. And, this make it difficult to access the information needed.

Supposed if we would like to find the document about World Cup 2002 which is hosted by Japan and Korea, there is no easy way to retrieve the right document containing the information we need as many existing search engines only retrieve documents crawled recently. In 1996, Internet Archive [9] is emerged as an organization providing the archived collections of web pages. They have implemented the search system called Way Back Machine which allows a user to search historical documents. This search functionality can be referred as temporal search. The inconvenience of this system is that the user must enter the URL and the date of web page he/she want to search for.

The Way Back Machine search engine is considered not so useful in practice because it is rather bothering for users to provide the system with time period, but specify only search keywords. The system should have an ability to automatically retrieve results and rank them based on time of the document. This comes to the question of how to determine the time of document. Many web documents contain timestamps that may not always trustable. Inability to identify timestamp of document prohibits the performance of temporal search. Not only in the Information Retrieval needs to extract temporal information from documents, but also in the Emerging Trend Detection (ETD) research [3]. EDT is focused on discovery the emerging trend topics from text documents where the event time of each document has to be explicitly assigned.

The main purpose of this paper is to discuss about the current works on an estimation of time from documents. The motivation in another word is to predict (or judge) the most likely originate time of document. In addition, we will provide a descriptive framework to compare existing works. In the following section, we will give an overview of research fields which will use as fundamental techniques in document time extraction task. The proposed descriptive framework will be presented in Section 3. Finally, in Section 9, we will give a summary and conclusion.

2 Background

2.1 Information Retrieval vs. Information Extraction

Information Retrieval (IR) provides the user with the ability to access to information needed. The IR field is originally based on the idea of question-answering system [2]. However, the system that retrieves documents from document collections to satisfy with the user's requirement is also called an IR system. In this case, we can refer to the document retrieval system as the IR system interchangeably. The basic processes in IR are generally composed of query interpretation, document representation, indexing and a ranking method for retrieved documents. Each process involves common techniques from linguistics, arithmetic, and statistics. For instance, to transform a search keyword to query language requires the knowledge from linguistics, or natural language processing (NLP). Moreover, how to compute the similarity of a document to the user's query can be performed using arithmetic or numeric methods (Boolean operations, an inner product, or a cosine distance). Finally, statistic approaches are widely used in document ranking process to present retrieved information in an order of relatedness.

Another field similar to IR is Information Extraction or IE. The goal of IE is to extract interesting information from documents for an automatic analysis by a computer. The extraction techniques have to deal with the understanding of the meaning of natural language. First, it is necessary to know what kind of semantic information interesting, and then identify a part of text and assign particular attributes to it. The fundamental steps in IE are parallel to those in IR where the text is tokenized into tokens, after that part-of-speech tagging is performed to obtain the syntactic role of each token in the sentence. The next step is to identify name entities, i.e. person names, organization names and locations. Those information identified will be used to form a sentence or phrase which semantically interpreted by the computer. Most of IE systems demand human experts to assist in some processes while recently IE researches employ data mining technique to avoid the cost of human involvement.

2.2 Text mining vs. Data mining

We now draw closer to the fields that will be used in temporal information extraction process. Text mining is the main process employed in the task. Text mining refers to the process of extracting useful information and knowledge from unstructured text, or sequence of character strings [8]. Text mining is viewed as a new emerged field from diverse research areas, namely, information retrieval, machine learning, statistics, computational linguistics, and data mining. The difference between text mining and data

mining is that text mining requires preprocessing technique to convert unstructured data into structured format in which is suitable for mining algorithm, whereas data mining concerns on the discovery of knowledge from structured data such as the database. Thus, data preprocessing, data modeling as well as an adaptation of existing mining algorithm, are mainly interested in text mining field.

There are three most common patterns of knowledge can be discovered by using text mining: a concept distribution, frequent or near frequent sets, and an association [3]. The concept distribution, can be used to find a concept in the document where the distribution proportion of concept is high compared to overall distribution. The frequent set is a concept often occurred in the document which can be found by Apriori algorithm [1] while the near frequent concept is an undirected relation between two frequent sets of concepts. The relationship can be measured by a degree of overlapping. The last pattern is an association which denotes a direct relation between two concepts or a set of concepts. Two parameters are used to determine an association rule: support and confidence. Support indicates how often both concepts are occurred together while confidence tells the proportion of co-occurrence of both concepts to the occurrence rate of another concept.

2.3 Classification and Clustering

Text categorization is a process of assigning the correct topic to each document. For example, the news collection can be classified into the topics like “politics”, “sport” or “technology”. There are two possible manners of text categorization: classification and clustering. The first method relies on the predefined topics (categories) and a training process called *supervised learning* while the latter method, *unsupervised learning*, automatically group documents into topics. Both methods have to select index terms or an appropriate set of keywords representing the content of documents. The set of index terms can be modeled as a bag-of-words, controlled vocabularies or feature vectors [20]. The data preparation procedure will be done in the same manner as a document preprocessing step in IR starting from tokenization, stopword removal, stemming and lemmatization. Each term in the document should not be equally important. Hence, the term weighting scheme namely TF-IDF[19] is needed to diminish less useful terms. Moreover, another filtering process should be performed in order to select only significant terms to make categorization efficient. Two methods for feature selection as follows:

- Select features based on their computed entropy [14]. The term entropy, or noise measure gives a value of how well a term is suited for separate a documents from other documents in the document collection. The term that occurs in few documents will have higher entropy compared to the one appearing in most of documents. Therefore a high entropy term is suitable to distinguish the document from the rest.
- Another mean for choosing a feature is to use information gain [8]. Information gain is measurement of how much information we can obtain from a term where it takes into account the relationships between features and categories. The higher of information gain value, the more informative of the term.

The learning algorithms used by text categorization are existing techniques in machine learning. For supervised learning, the categories are predefined and the documents with

labels are used as a training set. The well-known classification algorithms [5] are Naïve Bayes Classifier, Nearest Neighbor Classifier, Decision Trees, Support Vector Machines and Kernel Methods. Unsupervised learning does not require any training set. The well-known clustering algorithms are Hierarchical Clustering, K-Means, Bi-Section-K-Means, Self Organization Map, EM-Based Clustering and Fuzzy Clustering.

2.4 Temporal Database and Indexing

Temporal database as defined by [16] is the repository of documents storing the history or old versions of documents. Hence, the database system is capable of retrieving any particular version of a document. In [16], they proposed the methodology to organize documents and how to incorporate temporal information in document indexing. Their experiments proved that the system yielded a very high performance concerning temporal related queries. The similar work is presented in [10]. The idea is to support time-travel text search by underlining on space and time efficiency. First, the mechanism called temporal coalescing is introduced to reduce the size of inverted file index with data accuracy tradeoff. Second, sublist materialization, or the index accessing technique is operated on conflated index while the system still has a good speed.

3 Temporal Information Extraction Process

In this section, we proposed the descriptive framework of temporal information extraction. The framework consists of two main parts as follows:

- The standard processes of temporal information extraction model
- Subprocesses (or tasks) performed in each phrase of main processes

The first part of our framework demonstrates essential processes that have to be implemented in all temporal information extraction works. Each main process is decomposed into subprocesses. Those subprocesses will be executed in order to accomplish the main process.

3.1 Main processes

The common processes for predicting time of the document comprises of 5 phrases:

Document preprocessing: to prepare documents for further processing

Data modeling: to encode data into a proper data model

Information extraction: to employ an extracting algorithm for temporal information

Evaluation: to verify an accuracy of results obtained

Data presentation: to display temporal information by using visualization

The five main phrases are depicted in Figure 1. First, it starts with *data preprocessing* step. Input is a document from the document collection. Normally the document is represented in text format, or a sequence of strings. Before it can be used, we have to do document preprocessing to transform unstructured data into structured data. There

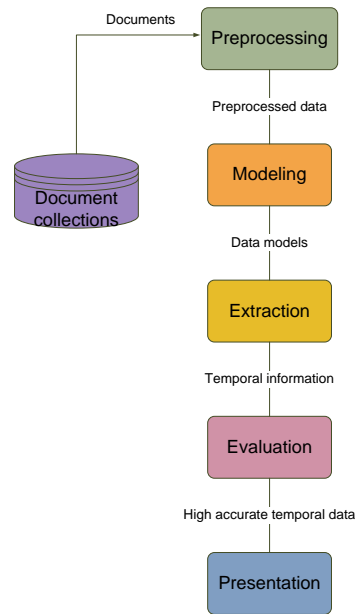


Fig. 1. Five Main Processes

are many specific steps involve which are similar to those techniques used in IR and IE fields. Some applications depend on complicated methods from natural language processing, or some only apply a simple data preprocessing step in order to extract data from texts. After preprocessing of raw data, the system performs *data modeling* to encode data into a data model which will become suitable for further processing. At this step, it generates two model views: document view, and temporal view. Similar to document preprocessing, document modeling are also adapted from the models used in an IR system. We will discuss in more details about data preprocessing and modeling in Section 3.2.

The 3rd phrase is *temporal information extraction*. We have to choose the appropriate extracting algorithm to discover temporal information from documents. The algorithms presented in exemplar works are methods deviated from the fields of machine learning, statistics, or IR with the goal of finding useful patterns. The important thing is that mining or learning algorithms selected have to match with data models they used in the previous step. Next step called *evaluation*, the system will assess extracted results to decide if that outcome can be used or not. Predicting the time of document is similar to document classification where the document is classified into the most likely originated time period [4]. Consequently, the system should be able to determine how much confidence it has for classifying documents. This is rather important issue for statistical tasks because confidence measure indicates the reliability of a system. There are various ways to evaluate results employed in temporal information extraction works.

We will explain them later. The final step is to display results with high confidence. *Visualization* is one technique that can be used to display temporal information.

Each of those steps described above can be viewed as a high level process. It is possible to break down the main process into subprocesses or tasks which must be performed to complete main processes.

3.2 The Structure of Processes and Methods

As we have discussed in the last section, the main process will be decomposed into subprocesses, or tasks to achieve its goal. Tasks, or in another word, detailed mechanisms are obligatory procedures in each main process. However, each task can be performed by deploying one or more methods. Our proposed analysis framework will be based on comparison of methods carried out in each task. The task-method diagram is shown in Figure 2. Subprocesses or tasks are linked with a main process using *a normal line with arrow* indicating the decomposition of process in high-level where each subprocess must be executed. In contrast, another kind of links, *a dash line without arrow* connects each subprocess with potential methods that can be applied to finish the subprocess.

In the second level there are five major tasks for the top process in Figure 1. All of these five tasks must be performed in order to accomplish a process, or task in upper-level. Recursively, each task in the second level can be separated into smaller tasks. For example, in document preprocessing task, it is composed of tokenization, tagging, stop-word removal, lemmatization, stemming, and feature selection. All lower-level tasks need to be done in according to complete the document preprocessing task. The links between a task and methods (dash line) identify the possible methods for solving the task, e.g. there are two ways for tagging task: part-of-speech tagging or time tagging. Those methods are alternatively chosen by a particular system and they are equivalent to procedures used to access, operate data, and execute tasks. Using only one method or the combination of several methods is possible depending on domains of applications.

In the next five sections, we will explain and compare the methods used by exemplar works. The analysis we performed is not systematic review, nonetheless, it gives details of outstanding mechanisms implemented by the six systems JRH05[4], LBA01[13], SJ00[23], MW00[15], SA00[22], SA99[21] and AG06[17] as well as comparison among alternative methods.

4 Document preprocessing

In general input data is the collection of documents which can be domain-specific, or domain-independent, structured or non-structured text, and monolingual or multilingual documents. There are some reference collections available for controlled comparison, such as, TREC collection, Time, REUTER or TDT corpus. Those collections provide a standard input data with semi-structure format. However, a collection of articles and newspapers is not update-to-date and gathered from very short period which is unsuitable for temporal information task. Hence, most of document collections used in sample

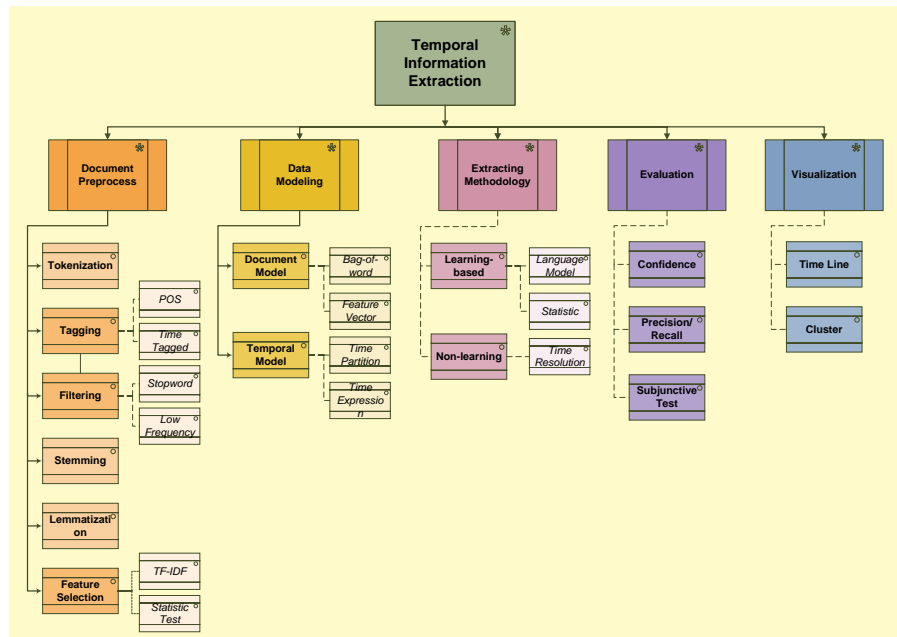


Fig. 2. A Diagram of Processes and Mechanisms

systems are manually collected from diverse sources and mainly in the format of non-structured text. The document collection used in JRH05 is gathered from three Dutch newspaper sites, LBA01 analyzed 4 newspaper sources while SJ00, SA00 and MW00 used the data from TDT project.

The document preprocessing step is analogous to that in IR and IE tasks. In the leftmost of Figure 2, we can see an overall steps of transforming documents into useful data format.

1. Tokenization splits the text into a list of words, or tokens. In English, we use sentence delimiters, i.e. a period, a question mark, an exclamation mark, or a comma. However, there are some problems and exceptions of tokenization [7].
2. Tagging is the process of labeling each token with interesting information. There are two possible types of tagged info: part-of-speech (POS) tagging and time tagging. The first method is aimed to mark a token with its part-of-speech in the sentence, such as, a noun, a proper noun, an adjective, a verb, a determiner, or a number. POS tagging helps in removal of irrelevant words (adjectives, or determiners) and also can reduce ambiguity of word senses with several meanings (a noun or a verb). Besides, tagged tokens will be useful for stemming process. SA00 and SA99 used a POS tagger program implemented by University of Massachusetts. Moreover, they also used a shallow parser called Badger to find noun phrases and name entities[6].

The second type of tagging methods is time tagging. Not only a syntactical tagging has been done, but some systems also tag the document with time expressions. For

example, in LBA01 work, they performed a shallow syntactic parser and lexical analysis to find time related words (cardinal or ordinal numbers, day of weeks, months, seasons, holidays, recurring events, time prepositions, or conjunctions). After identifying of temporal words, the system assigned a date to each temporal word which can be an absolute date or a relative date. Phrases as “4th of July, 2000” and “Valentine day of this year” are possibly assigned the absolute dates whereas, “today” or “next week” will be given relative dates. The absolute date is represented using a time point while the relative date can be expressed by a time interval, or time span. More details of time definition will be explained in Section 5.2 Similar to LBA01, MW00 used time tagging method.

3. Filtering intends to remove words with less information or without. The general word filtering method is stopword removal which aims to eliminate less informative or useless words for indexing. The highly frequent words like articles, prepositions, and conjunctions are stopwords which are not useful in distinguish among documents. Another direct way is to simply neglect words with less than a certain number in overall corpus. JRH05 used a cut off value 10 where they considered a good threshold for deleting a large portion of misspelling words.
4. Stemming process decreases syntactic variation of words by transforming into a common form (a root of word, or stem) for example, ‘cars’ becomes ‘car’. Stemming helps in reducing a huge amount of words. The easy and efficient method for stemming is to do affix removal by writing rules, e.g. ($sSES \rightarrow ss|s \rightarrow \phi$). In JRH05, they applied Dutch rule-based stemming algorithm which cut down the size of vocabulary in a great quantity.
5. Lemmatization is the lexicon approach mapping inflections of words into one canonical representation or lemma, such as, mapping various verb forms to infinite, mapping plural noun to singular form, or mapping comparative forms of an adjective to the normal form. There are several strategies to implement lemmatization depending on a language (Russian and Polish expand with all full forms, German and French expand with base form, or Japanese and Korean replace with base form).
6. The last step of document preprocessing is feature selection. After we reduce a lot of words from document collections. Nevertheless, those remaining words are still in a large number. So, we have to do further process for decreasing vocabulary size. The well-known feature selection is to weigh a term with TF-IDF score where TF is a term frequency and IDF is an inverse document frequency. After computing TF-IDF score, the top rank N terms will be selected as index terms. In addition to this approach, we can weigh a term using Information Gain or Term Entropy as described in Section 2.3

In SA99 and SA00, they used statistic approach to select term or significant features. Based on the assumption that the distribution of a feature depends on a base rate of occurrence that does not vary with time, they employed hypothesis testing to determine the most important terms. Statistic testing is compared against a threshold (p) for determining if a feature is significant or not. In their systems, the p value is set to 0.05 meaning the significant feature must be seen at least 18 days in a year. Note that the term mostly occurs is uninteresting, and the most interesting term has prominent distribution within sample period.

5 Data Modeling

Before we apply mining algorithms on data, we have to encode it into a data model which will become suitable for processing. Data modeling step consists of 2 operations: 1) modeling a document, and 2) modeling a temporal view.

5.1 Modeling A Document

Similar to IR models, the document can be represented in two patterns, bag-of-words and feature vector. In the first type, a document is modeled by an unordered list of terms where grammatical information is absent. The latter type treats each document as a vector of terms in the vector space. Different weighting schemes like binary or TF-IDF can be applied.

5.2 Modeling A Temporal View

Each document will have time associated to it. The simple method of modeling temporal view of the document is to partition time span into smaller time periods. The time span is divided according to time granularity, e.g. daily, weekly, and monthly, etc.

The more complicated temporal model is presented in LBA01. A time point (T) is a sequence of granularities (g) and positive integers (n) : $T = g_1n_1g_2n_2 \dots g_kn_k$. Each integer n_i must be a valid number of granularity g , for example, if the granularity is month, the number n , or a date can be 0 to 31. Hence, the date as “4th of July, 2000” and “Valentine day of this year” are represented in a time point format as ‘y2000m7d4’ and ‘y2008m2d14’ sequentially. The time interval is a certain period of time between two time points; $I = [T1, T2]$ where $T1$ and $T2$ have the same time granularity and $T1 < T2$. The span of time is unanchored directed interval of time: $S = \pm T$. The sign in front of a time period could be \pm where it indicates the direction of time span (+ shifts to the future, - shifts to the past). Thus, for those temporal related words unable to obtain absolute dates will be assigned relative dates in the form of a time interval or a time span. In this case, “next week” and “from June to July” will be expressed with relative times like ‘+7d’ and ‘[m6, m7]’. LBA01 gave a definition of the *temporal expression* as an expression of a time point, and a time operator. The time operator is a function operating on time points, for instance, $shift(T_k, S)$ operator changes a time point T_k to a new time point T_{k+1} by a time period S (a number follows by the granularity). $shift(y2000m7d4, +7d)$ means shifting from the date y2000m7d4 to next 7 days, or next week. Finally, the system will tag document with absolute date time, or time expressions of absolute times and relative times. Similar to LBA01, MW00 has defined the tag **TIMEX** (stand for Time Expression). Time is represented using the ISO standard format $CC : YY : MM : DD : HH : XX : SS$ as a value of the particular time point (in a calendar coordinate system), and a week, or a day of the week also can be expressed in $CC : YY : Wwd$ format, e.g. “Tuesday, 1st of January 2008” and “last week of last year” are represented with ‘20 : 08 : 01 : 01 : 00 : 00 : 00’ and ‘20 : 07 : W52’. Tagging method uses the reference time to help resolve context-dependent expressions. The reference time is assigned the value of either the Temporal Focus or the document (creation) date. The Temporal Focus is the time currently being

talked about in the narrative. The initial reference time is the document date. They assigned two different types of time value: generic and specific one, where specific time is assigned a value based on an offset from the reference time, but the non-specific, or generic time i.e. “nowadays”, “today” in the context “a new generation today”, “Winter” in “Winter is usually full of snow” is not assigned time value. The difference between temporal model of LBA01 and MW00 is that MW00 didn’t define any operations on time. As we see that the systems have to deal with natural language processing on both syntactic and semantic issues. Linguistic and lexical analysis is required to be able to identify a time from explicit and implicit information, resolve ambiguous time using time expression, and finally tag document with unambiguous time.

In JRH05 system, they have chosen simple methods for both document and temporal modeling. A function $Time(d_i)$ is defined to determine the date of documents. Moreover, data model is composed of document and temporal model as follow:

The document collection contains a list of corpus documents defined as follows:

$$C = \{d_1, d_2, d_3, \dots, d_n\} \quad (1)$$

Each document has two attributes, time span and indexing features.

$$d_i = \{(t_i, t_{i+1}), \{f_1, f_2, f_3, \dots, f_n\}\} \quad (2)$$

$$\text{where } t_i < t_{i+1} \text{ and } t_i < Time(d_i) < t_{i+1}$$

The granularity of model is depended on the time span of corpus collection. For example, if we want to predict document time in a particular year, we must partition document time span on yearly basis. JRH05 designed two data structure for keeping document models and temporal views as shown in Table 1. Both data structures have different pro and con. A table is good when data is sparse, and efficient in sorting. On the other hand, a matrix gives a direct access to data which can improve access time.

6 Extracting Methodology

The next main process is to extract temporal information from data we have. We categorize algorithms for extraction into two classes: learning-based algorithm and non-learning one.

1. Learning-based approach

Statistic language model is a method based on learning algorithm employed in JRH05 system. They used the temporal language model for dating task. The temporal language model, or time-based model [12], is the variant of probabilistic model [18] one of prominent IR models. The nature of language has temporal metadata in itself. For instance, a certain topic or event is likely to occur in a particular period. One would expect to see the news about tsunami in South-East Asia after the date 24th of December 2004. Hence, we can say that the temporal language model assigns a probability to a document according to word usage statistic over time. JRH05’s work used a normalized log-likelihood ratio measure proposed by [11] to

Table 1. Data Models: Table vs. Matrix

(a) Table

Word	Partition	Frequency
terrorist	2002	9478
terrorist	2003	7750
terrorist	2004	5212
tsunami	2002	101
tsunami	2003	56
tsunami	2004	26905
world cup	2002	19273
world cup	2003	6069
world cup	2004	448

(b) Matrix

	2002	2003	2004
terrorist	9478	7750	5212
tsunami	101	56	26905
world cup	19273	6069	448

compute to language models. Given a data model of corpus documents as shown in Table 1(a), if we want to date an unknown-date document D , we have to compare the language model of D with those of corpus documents with the following equation:

$$Score(D, P) = \sum_{w \in D} \Pr(w|D) \times \lg \frac{\Pr(w|P)}{\Pr(w|C)} \quad (3)$$

where C is the background model estimated on entire collection

To avoid undefined value of $\lg \frac{\Pr(w|P)}{\Pr(w|C)}$ in a formula above, they suggested to assign a very small (non-zero) probability value, called *smoothing method*. The computed score indicates the probability of the document given the time partition. Consequently, the unknown-date document will be classified into the time partition most likely to its originated time.

In SA1999, they used the same technique for feature selection, known as *statistic test*, to estimate time for the group of feature. The test is performed on two significant features having overlapped time period to determine if they are statistically related. If the computed value from testing is above a threshold, the two features are coalesced into a single topic. Tagged time of both features will be use as the story time.

2. Non-learning approach

LBA01 and MW00 did not rely on learning method for extracting temporal information. However, they used *time reference resolution* which rather simply resolved

time from time-tagged information. In LBA01, every time entity tagged in the document must be resolved into a concrete date. This can be done by looking at relations between time entities and reference time, and then executing time operations. For example, the sentence likes ‘It was yesterday.’ tagged with ‘ $-1d$ ’ will be given a concrete date by performing a shift operation on the publication date of the document. More details of resolution of temporal references can be found in LBA01. After the document is tagged with concrete dates, they computed the relevancy of date by frequency of appearance in the document. The most relevant date is used as the reference date for the document, however, if all dates are similar relevant, the publication date will be used instead. Then, the event-time period of the document is generated by assembling all nearby dates to the reference date where their relevancy must be greater than a threshold.

In the same way, MW00 required time tagged data before temporal information extraction could be performed. However, MW00 is slightly different from what have been implemented in LBA01. They proposed two steps for chronology extraction. First, they tried to find temporal relations between events called event ordering. Each event is associated to a verb which will be encoded using a tuple $\langle ST, Rel, RT, Rel, ET \rangle$ where ST , RT and ET are a speech time, reference time and event time accordingly. Temporal relations could be precede, follow, or coincide. Event ordering step helps in resolving a value for generic time defined in Section 5.2 by either shifting or maintaining from found temporal relations. The next step conducted by MW00 is event time alignment. It is the process of locating events on a calendrical line from the TIME expressions tagged by previous step. All verbs that lack temporal expressions will be assigned the values of recently appeared time expressions where the verbs such as auxiliaries, modals and verb following ‘to’, ‘not’ or specific modal verb are in contrast. Their time values will be given to the immediately preceding verbs missing time expressions. The authors of MW00 claimed that this method is rather “blinding propagating of time expressions based on event proximity”. They would like to consider also temporal coordinator words and explicitly temporal adverbial phrase. In MW00, they didn’t state clearly about the mechanism to judge the date for representing a document.

7 Evaluation of Results

In JRH05, before giving the particular date for a document, it has to measure how confident it is to suggest. They simply obtained confidence value from computing the relative distance between score of the top ranked time partition to the score of the following one. The distance value should be greater than a user-defined threshold, confidence value. It is preferred by the system dealing with classification tasks to gain a very high precision. The following is the formula for confidence measurement used by JRH05.

$$Conf(Time(D)) = -\lg \frac{score(P_1)}{score(P_2)} \quad (4)$$

where $Time(D)$ is the dating function of the document D

P_1 , P_2 are the first, second ranked time partition consecutively. JRH05 evaluated their system performance by calculating the percent of correctly dated documents with five different confidence values. The evaluation method presented in LBA01 is precision and recall where precision means the valid extracted dates per total extracted dates, and recall means valid extracted dates per valid dates in the collection.

SA99 has proposed a subjunctive assessment. They tested the system again Fact on File. First, features are relevant if they appeared in Fact on File with overlapping dates. Second, the generated stories are relevant if their times overlap with the stories being considered and also if they have a least one term in common. After all, they calculated precision/recall from accessed results.

8 Visualization

There are only two systems that implemented visualization of their results. First in SJ00, they displayed temporal information using a time line. The x-axis represents time dimension while the y-axis indicates the importance of topic. For each news topic, it will be shown in the time-axis according to temporal information associated to it. And, the most important topic will be placed on a higher position compared to the less important one. Figure 3 portrays the visualization in SJ00.

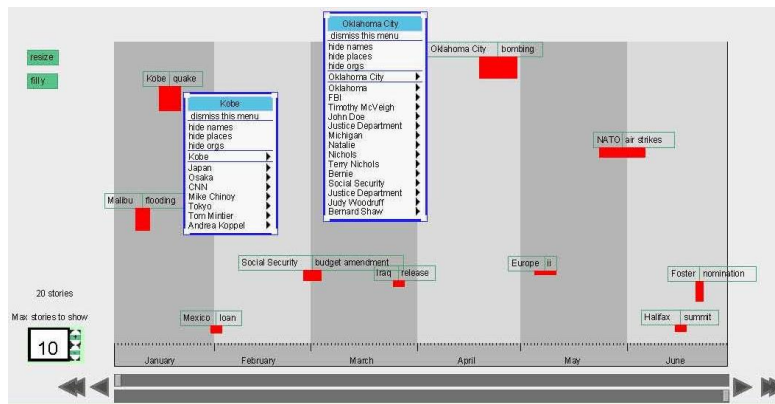


Fig. 3. Time Line Visualization[23]

In AG06, they proposed a prototype of visualization using clustering technique based on temporal attributes. Those attributes are referred as a date and time reference in a document. A hit list of documents obtained from the search engine will be clustered into a time-related group. For instance, if a user search using 'world cup' keyword, the system will retrieve all documents related "world cup" including every tournament in the past. Besides, retrieved documents will be grouped into the year of the world cup event before present to the user. An example of clustering results is depicted in Figure 4.



Fig. 4. Clustering of Temporal Data[17]

9 Summary

Time becomes an interesting dimension to be explored from the document. Each document is associated with temporal information in both explicit and implicit forms. Explicit temporal data can be represented by metadata (e.g. a created date, or a published date) while implicit temporal data is embedded in the content of document itself. Many research works are conducted on extracting temporal information from documents. We can conclude that there are only two approaches used in extracting process. The first method relies on a learning algorithm and temporal data will be draw from the patterns found in document collections. Another method applies NLP techniques in performing syntactic and semantic analysis of the document. This approach tries to understand the meaning of the document content and generate temporal information.

Several research fields are focusing on time aspect, namely, database, information retrieval, or emerging trend detection (ETD). Thus, there are a lot of possibility and challenge in this area especially in Temporal Search application. In addition to finding temporal information in the document, other potential research topics are as follows:

- How to organize indexing of documents with temporal information to be efficient in accessing and retrieval
- How to rank and present documents using temporal information according to the user's need
- How to handle the variant of language when a query term or terms in the document are obsolete

References

1. R. Agrawal, T. Imielinski, and A. Swami. Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, 5(6), 1993.
2. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.
3. M. W. Berry. *Survey of text mining : clustering, classification, and retrieval*. Springer-Verlag New York, Inc., 2004.
4. F. de Jong, H. Rode, and H. D. Temporal language models for the disclosure of historical text. In *Proceedings of AHC'2005 (History and Computing)*, 2005.
5. R. Feldman and J. Sanger. *The Text Mining Handbook*. Cambridge, 2007.
6. D. Fisher, S. Soderland, F. Feng, and W. Lehnert. Description of the umass system as used for muc-6. In *MUC6 '95: Proceedings of the 6th conference on Message understanding*, pages 127–140, Morristown, NJ, USA, 1995. Association for Computational Linguistics.
7. G. Grefenstette and P. Tapanainen. What is a word, what is a sentence? problems of tokenization. In *Proceedings of the 3rd Conference on Computational Lexicography and Text Research*, 1994.
8. A. Hotho, A. Nürnberger, and G. Paass. A brief survey of text mining. *LDV Forum*, 20(1):19–62, 2005.
9. Internet Archive. <http://archive.org/>.
10. T. N. Klaus Berberich, Srikanta J. Bedathur and G. Weikum. A time machine for text search. In *SIGIR*, 2007.
11. W. Kraaij. Variations on language modeling for information retrieval. *SIGIR Forum*, 39(1):61, 2005.
12. X. Li and W. B. Croft. Time-based language models. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, 2003.
13. D. M. Llidó, R. B. Llavori, and M. J. A. Cabo. Extracting temporal references to assign document event-time periods. In *Proceedings of DEXA'2001*, 2001.
14. K. E. Lochbaum and L. A. Streeter. Comparing and combining the effectiveness of latent semantic indexing and the ordinary vector space model for information retrieval. *Inf. Process. Manage.*, 25(6):665–676, 1989.
15. I. Mani and G. Wilson. Robust temporal processing of news. In *ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, 2000.
16. K. Nørsvåg. Supporting temporal text-containment queries in temporal document databases. *Journal of Data & Knowledge Engineering*, 49(1):105–125, 2004.
17. M. G. Omar Alonso. Clustering of search results using temporal attributes. In *Proceeding of the 29th SIGIR*, 2006.

18. J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR*, pages 275–281, 1998.
19. G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, 1988.
20. G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.
21. R. Swan and J. Allan. Extracting significant time varying features from text. In *Proceedings of CIKM '99*, 1999.
22. R. Swan and J. Allan. Automatic generation of overview timelines. In *Proceedings of the SIGIR'2000*, 2000.
23. R. Swan and D. Jensen. Timemines: Constructing timelines with statistical models of word usage. In *Proceedings of KDD-2000 Workshop on Text Mining*, 2000.