# Improving Temporal Language Models For Determining Time of Non-Timestamped Documents

Nattiya Kanhabua and Kjetil Nørvåg

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

**Abstract.** Taking the temporal dimension into account in searching, i.e., using time of content creation as part of the search condition, is now gaining increasingly interest. However, in the case of web search and web warehousing, the timestamps (time of creation or creation of contents) of web pages and documents found on the web are in general not known or can not be trusted, and must be determined otherwise. In this paper, we describe approaches that enhance and increase the quality of existing techniques for determining timestamps based on a temporal language model. Through a number of experiments on temporal document collections we show how our new methods improve the accuracy of timestamping compared to the previous models.

## 1 Introduction

During the recent years, the amount of information on the Internet has increased dramatically, and makes web search even more challenging. Although well-known search engines still deliver good results of pure keyword searches, it has been observed that precision is decreasing, which in turn means that a user has to spend more time in exploring retrieved documents in order to find those that satisfy the information need. One way of improving precision is to include the temporal dimension into search, i.e., extending keyword search with the creation or update time of the web pages/documents. In this way, the search engine will retrieve documents according to both text and temporal criteria, i.e., *temporal text-containment search* [14]. In addition to searching the current web, searching in old versions of web pages is sometimes useful. This can be of interest in large-scale archives like the Internet Archive [5] as well as more focused web warehouses like V2 [13].

However, in order for temporal text-containment search to give good results, it is obvious that the timestamps of documents have to be as accurate as possible. In the case of local document archives, trustworthy metadata that includes time of creation and last update is available. However, in the case of web search and web warehousing, having an accurate and trustworthy timestamp is a serious challenge. One way to solve the problem, is to use the time of discovery as timestamp (i.e., the time a document/web page is first found by the web crawler). This will give an accurate timestamp if the creation time of a document and the time when it is retrieved by the crawler coincide in time. Unfortunately there is no guarantee that this is the case, and adding to the problem is the fact that the web page/document can be relocated and discovery time in this case will be very inaccurate. In some cases metadata about documents on the web can be retrieved but they can also in general not be trusted and often are simply just plain wrong.

As can be seen, in the case of web search and web warehousing it will in general be impossible to get trustworthy timestamps based on information acquired during crawling time. Thus, our research challenge is: for a given document with uncertain timestamp, can the contents of the document itself be used to determine the timestamp with a sufficient high confidence? To our knowledge, the only previous work on this topic is the work by de Jong, Rode, and Hiemstra [3], which is based on a statistic language model. In this paper, we present approaches that extend the work by de Jong et al. and increases the accuracy of determined timestamps.

Our main contributions in this paper are 1) a semantic-based preprocessing approach that improves the quality of timestamping, 2) extensions of the language model and incorporating more internal and external knowledge, and 3) an experimental evaluation of our proposed techniques illustrating the improved quality of our extensions.

The organization of the rest of the paper is as follows. In Section 2, we give an overview of related work. In Section 3, we outline preliminaries that will be used as the basis of our approach. In Section 4, we explain semantic-based techniques used in data preprocessing. In Section 5, we propose three new approaches that improve the previous work: word interpolation, temporal entropy and using external search statistics. In Section 6, we evaluate our proposed techniques. Finally, in Section 7, we conclude and outline future work.

## 2 Related Work

To our knowledge, there is only a small amount of previous work on determining time of documents. This aim can be divided into two categories: determining time of creation of document/contents, and determining time of topic of contents. For example, a document might be created in 2002 but the contents is about the Viking Age.

Determining time of a document can be done using 2 techniques: learning-based and non-learning methods. The difference between the two methods is that the former determines time of a document by learning from a set of training documents, while the latter does not require a corpus collection. Learning-based methods are presented in [3, 17, 18]. In [17, 18], they use a statistical method called *hypothesis testing* on a group of terms having an overlapped time period in order to determine if they are statistically related. If the computed values from testing are above a threshold, those features are coalesced into a single topic, and the time of the topic is estimated from a common time period associated to each term. Another method presented by de Jong et al. in [3] is based on a statistic language model where time of the document is assigned with a certain probability. We will discuss in details this statistic language model in the next section.

Non-learning methods are presented in [9, 11]. They require an explicit time-tagged document. In order to determine time of a document, each time-tagged word is resolved into a concrete date and a relevancy of the date is computed using the frequency of which the date appears in the document. The most relevant date is used as a reference date for the document, however, if all dates are similar relevant, the publication date will be used instead. In the end, the event-time period of the document is generated by assembling all nearly dates to the reference date where their relevancy must be greater than a threshold.

Comparing the non-learning to learning-based methods, both of them return two different aspects of time. The first method gives a summary of time of events appeared in the document content, while the latter one gives the most likely originated time which is similar to written time of the document.

Also related is work on indexing, retrieval, ranking and browsing. Recent work on indexing and retrieval include the work on the V2 system [13, 14]. A technique for indexing and ranking is described in [2]. In [1, 15], Alonso et al. present an alternative document ranking technique that uses temporal information to place search results in a timeline, which is useful in document exploration/browsing.

## 3 Preliminaries

In this section, we briefly outline our document model and the statistic language model presented by de Jong, Rode and Hiemstra [3]. For short we will in the following denote their approach as the *JRH approach*.

### 3.1 Document Model

In our context, a document collection contains a number of corpus documents defined as $C = \{d_1, d_2, d_3, \ldots, d_n\}$. A document has two views: a logical view and a temporal view. The logical view of each document can be seen as bag-of-word (an unordered list of terms, or features), while the temporal view represents trustworthy timestamps. A simple method of modeling the temporal view is partitioning time spans into a smaller time granularity. A document model is defined as $d_i = \{\{w_1, w_2, w_3, \ldots, w_n\}, (t_i, t_{i+1})\}$ where $t_i < t_{i+1}$, $t_i < Time(d_i) < t_{i+1}$, and $(t_i, t_{i+1})$ is the temporal view of the document which can be represented by a time partition. $Time(d_i)$ is a function that gives trustworthy timestamp of the document and must be valid within in the time partition.

### 3.2 The de Jong/Rode/Hiemstra Temporal Language Model

The JRH approach is based on a statistic language model for timestamp determination. This *temporal language model* is a variant of the time-based model in [8], which is based on a probabilistic model from [16]. The temporal language model assigns a probability to a document according to word usage statistics over time. In JRH a normalized log-likelihood ratio [7] is used to compute the similarity between two language models. Given a partitioned corpus, it is possible to determine the timestamp of a non-timestamped document $d_i$ by comparing the language model of $d_i$ with each corpus partition $p_j$ using the following equation:

$$Score(d_i, p_j) = \sum_{w \in d_i} P(w|d_i) \times \log \frac{P(w|p_j)}{P(w|C)} \tag{1}$$

where $C$ is the background model estimated on the entire collection and $p_j$ is a time partition. The timestamp of the document is the time partition which maximizes the score according to the equation above. The intuition behind the described method is that given a document with unknown timestamp, it is possible to find the time interval that mostly overlaps in term usage with the document. For example, if the document contains the word “tsunami” and corpus statistic shows this word was very frequently used in 2004/2005, it can be assumed that this time period is a good candidate for the document timestamp.

As can be seen from the equation, words with zero probability are problematic, and smoothing (linear interpolation [7] and Dirichlet smoothing [19]) is used to solve the problem by giving a small (non-zero) probability to words absent from a time partition.

## 4 Semantic-based Preprocessing

Determining timestamp of a document from a direct comparison between extracted words and corpus partitions has limited accuracy. In order to improve the performance, we propose to integrate semantic-based techniques into document preprocessing. We have in our work used the following techniques:

- **Part-of-Speech Tagging:** Part-of-speech (POS) tagging is the process of labeling a word with a syntactic class. In our work, we use POS tagging to select only the most interesting classes of words, for example, nouns, verb, and adjectives.
- **Collocation Extraction:** Collocations [12] are common in natural languages, and a word can not be classified only on the basis of its meaning, sometimes co-occurrence with other words may alter the meaning dramatically. An example is “United States” as one term compared to the two independent terms “united” and “states”, which illustrates the importance of collocations compared to single-word terms when they can be detected.
- **Word Sense Disambiguation:** The idea of word sense disambiguation (WSD) is to identify the correct sense of word (for example, two of the senses of “bank” are “river bank” and “money bank”) by analyzing context within a sentence.
- **Concept Extraction:** Since a timestamp-determination task relies on statistics of words, it is difficult to determine timestamp of a document with only a few words in common with a corpus. A possibility is to instead compare concepts in two language models in order to solve the problem of less frequent words.
- **Word Filtering:** A filtering process is needed to select the most informative words and also decrease the vocabulary size. In our work, we apply the *tf-idf* weighting scheme to each term and only the top-ranked $N_t$ terms will be selected as representative terms for a document.

## 5 Enhancement of Statistic Language Models

In this section, we propose three new methods for improving the JRH approach: 1) word interpolation, 2) temporal entropy, and 3) external search statistics from Google Zeitgeist [4]. Each method will be described in more details below.

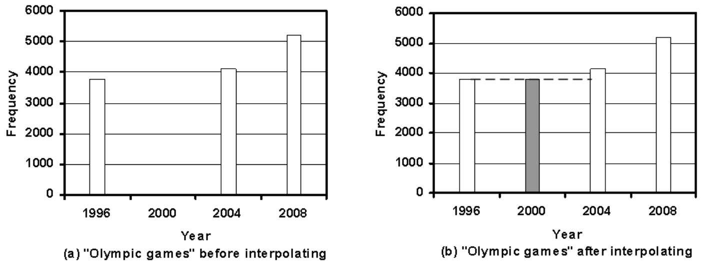


**Fig. 1.** An interpolation method for a recurring word

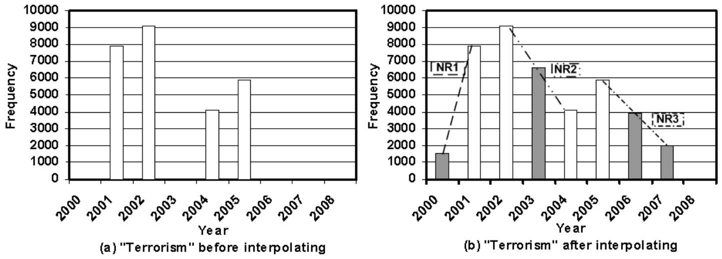


**Fig. 2.** An interpolation method for a non-recurring word

### 5.1 Word Interpolation

When a word has zero probability for a time partition according to the training corpus, this does not necessarily mean the word was not used in documents outside the training corpus in that time period. It just reflects a shortcoming of having a training corpus of limited size. As described in Sect. 3.2, smoothing can be used to model that a word also exists in other time partitions.

In the following we present more elaborate ways of word frequency interpolation for partitions where a word does not occur. In this process, a word is categorized into one of two classes depending on characteristics occurring in time: *recurring* or *non-recurring.* Recurring words are words related to periodic events, for example, “French Open”, “Christmas”, “Olympic Games”, and “World Cup”, and are supposed to appear periodically in time, for example December every year, or every four years. On the other hand, non-recurring words do not appear periodically (but might still appear in many time periods, and as such can be also classified as aperiodic).

How to interpolate depends on which category a word belongs to. All words that are not recurring are non-recurring, and thus it suffices to identifying the recurring words. This can be done in a number of ways, we initially use a simple technique just looking at overlap of words distribution at endpoints of intervals, for example when detecting yearly events look at all possible 12 month intervals (i.e., words on January 2000 and January 2001, February 2000 and February 2001. Note that the endpoints should actually be a bit flexible/wide, due to the fact that many events do not occur at the exact same date each year (Easter and Olympics are two typical examples).

Our interpolation approach is based on two methods: for recurring words, if they exist in a number of event periods those that are missing are automatically “filled in”, for non-recurring words interpolation is applied on periods adjacent to periods where the words exist.

**Recurring Words**: Assume a word $w_r$ that has been determined to be recurring, for example "Olympic Games". If the frequency of $w_r$ in a partition $p_j$, represented as $tf(w_r, p_j)$, is equal to zero, we interpolate $tf(w_r, p_j)$ with the minimum value of adjacent partitions, $\min(tf(w_r, p_{j-1}), tf(w_r), p_{j+1})$. As depicted in Fig. 1(a), the frequency is zero in the year 2000 (i.e., the word does not occur in any documents with timestamp within year 2000). After interpolating, Fig. 1(b) shows how the frequency in the year 2000 is assigned with that of 1996 because it is the minimum value of 1996 and 2004.

**Non-Recurring Words**: Assume a word $w_{nr}$ that has been determined to be non-recurring, for example "terrorism". Fig. 2(a) illustrates that a frequency is missing in the year 2000 because there is no event (occurrence of word) on "terrorism" in this year. On the other hand, in the year 2001 and 2002, "terrorism" becomes popular as terrorists attacked on $11^{th}$ of September 2001. Once again, information about "terrorism" is absent in the year 2003. However, "terrorism" becomes popular in the year 2004 and 2005 because of bombing in Madrid and London. Supposed, there is no major event on "terrorism" after the year 2005, so the frequency is zero in the year 2006, 2007 and 2008. Although the word does not occur in the corpus it is quite certain that the word still has been used in "the real world". We interpolate $tf(w_{nr}, p_j)$ in three ways.

In the case of a period $p_j$ where $w_{nr}$ has never been seen before, it is possible to observe $w_{nr}$ in that period. We interpolate $tf(w_{nr}, p_j)$ with a fraction (e.g. one-fifth) of $tf(w_{nr}, p_{j+1})$ where $p_{j+1}$ is the first partition $w_{nr}$ occurs. For example, the year 2000 is interpolated based on a fraction of the frequency in the year 2001. The interpolation method for this case is shown as *NR1* in Fig. 2(b).

In the case that $p_j$ is a period that $w_{nr}$ is supposed to be normally used, but is absent due to missing data, we interpolate $tf(w_{nr}, p_j)$ with the average frequency of the adjacent partitions, $\frac{tf(w_{nr}, p_{j-1}) + tf(w_{nr}, p_{j+1})}{2}$. For example, the year 2003 is interpolated with the average frequency of 2004 and 2005. The interpolation method of this case is shown as *NR2* in Fig. 2(b).

Finally, if $p_j$ is a period where $w_{nr}$ is absent because of decreasing popularity of the word, it can still be expected that $w_{nr}$ is used afterward, but not as much as before. We interpolate $tf(w_{nr}, p_j)$ with a fraction of $tf(w_{nr}, p_{j-1})$ where $p_{j-1}$ is the last partition $w_{nr}$ appears. In this case, the frequency of the years 2006, 2007 and 2008 are interpolated with a frequency of the year 2005 in a decreasing proportion. The interpolation method for this case is shown as *NR3* in Fig. 2(b).

### 5.2 Temporal Entropy

In this section we present a term weighting scheme concerning temporality called *temporal entropy* (TE). The basic idea comes from the term selection method presented in [10]. Terms are selected based on their entropy or noise measure. Entropy of a word $w_i$ is defined as follows:

$$Entropy(w_i) = 1 + \frac{1}{\log N_D} \sum_{d \in \mathbf{D}} P(d|w_i) \times \log P(d|w_i) \qquad (2)$$

where $P(d_j|w_i) = \frac{tf(w_i,d_j)}{\sum_{k=1}^{N_D} tf(w_i,d_k)}$, $N_D$ is the total number of documents in a collection $\mathbf{D}$ and $tf(w_i, d_j)$ is the frequency of $w_i$ in a document $d_j$. It measures how well a term is suited for separating a document from other documents in a document collection, and also it captures the importance of the term within the document. A term occurring in few documents has higher entropy compared to one appearing in many documents. Therefore, the term with high entropy, is a good candidate for distinguishing a document from others.

Similar to *tf-idf* but more complicated, term entropy underline the importance of a term in the given document collection whereas *tf-idf* weights a term in a particular document only. Empirical results showing that term entropy is good for index term selection can be found in [6]. Thus, we use term entropy as a term weighting method for highlighting appropriate terms in representing a time partition.

We define temporal entropy as a measure of how well a term is suitable for separating a time partition among overall time partitions and also indicates how important a term is in a specific time partition. Temporal entropy of a term $w_i$ is given as follows:

$$TE(w_i) = 1 + \frac{1}{\log N_P} \sum_{p \in \mathbf{P}} P(p|w_i) \times \log P(p|w_i) \tag{3}$$

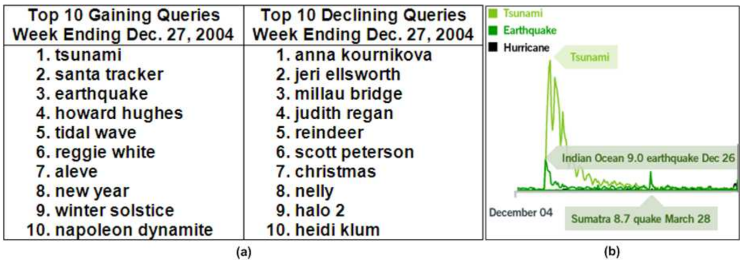
where $P(p_j|w_i) = \frac{tf(w_i,p_j)}{\sum_{k=1}^{N_P} tf(w_i,p_k)}$, $N_P$ is the total number of partitions in a corpus $\mathbf{P}$, and $tf(w_i, p_j)$ is the frequency of $w_i$ in partition $p_j$. Modifying the score in Equation (1), each term $w$ can be weighted with temporal entropy $TE(w)$ as follows:

$$Score_{te}(d_i, p_j) = \sum_{w \in d_i} TE(w) \times P(w|d_i) \times \log \frac{P(w|p_j)}{P(w|C)} \tag{4}$$

A term that occurs in few partitions is weighted high by its temporal entropy. This results in a higher score for those partitions in which the term appears.

### 5.3 Search Statistics

In our work, we have also studied how to use external knowledge, and in this section we describe how to make use of search statistics provided by a search engine. The only public available statistics that suits our purpose are those from Google Zeitgeist, which is given on different time granularities, such as week, month and year. We have employed the finest granularity available, i.e., weekly data. Fig. 3(a) shows a snapshot of search statistics which is composed of the top-10 rank for two types of queries. In the statistics, a query can be gaining or declining. A gaining query is a keyword that is growing in interest and becomes an emerging trend at a particular time. Fig. 3(b) shows the trend graph of the keywords “Tsunami” and “Earthquake”. Both words are gaining queries in December 2004 because they gain very high frequencies compared to a normal distribution and slightly decrease their popularity over the time line. In March 2005, the word “Earthquake” becomes a gaining query again because of an earthquake in Sumatra. On the other hand, a declining query is a keyword where its interest drops noticeably from one period to another.



| Top 10 Gaining Queries<br>Week Ending Dec. 27, 2004 | Top 10 Declining Queries<br>Week Ending Dec. 27, 2004 |
|---|---|
| 1. tsunami | 1. anna kournikova |
| 2. santa tracker | 2. jeri ellsworth |
| 3. earthquake | 3. millau bridge |
| 4. howard hughes | 4. judith regan |
| 5. tidal wave | 5. reindeer |
| 6. reggie white | 6. scott peterson |
| 7. aleve | 7. christmas |
| 8. new year | 8. nelly |
| 9. winter solstice | 9. halo 2 |
| 10. napoleon dynamite | 10. heidi klum |

(a)
🟩 Tsunami
🟩 Earthquake
■ Hurricane
Tsunami
Indian Ocean 9.0 earthquake Dec 26
December 04
Sumatra 8.7 quake March 28
(b)

**Fig. 3.** Google Zeitgeist: Search statistics and trends

By analyzing search statistics, we are able to increase the probability for a particular partition which contains a top-ranked query. The higher probability the partition acquires, the more potential candidate it becomes. To give an additional score to a word $w_i$ and a partition $p_j$, we check if $(w_i$,$p_j)$ exist as a top-ranked query. After that, we retrieve from statistics information about a query type (gaining or declining), query ranking and the number of partitions in which $w_i$ appears. Finally, a *GZ* score of $w_i$ given $p_j$ can be computed as:

$$GZ(p_j, w_i) = \Big(P(w_i) - f\left(R_{i,j}\right)\Big) \times ipf_i \tag{5}$$

where $ipf_i$ is defined as an inverse partition frequency and is equal to $\log \frac{N_P}{n_i}$. $N_P$ is the total number of partitions and $n_i$ is the number of partitions containing $w_i$. $P(w_i)$ is the probability that $w_i$ occurs; $P(w_i)$ = 1.0 if $w_i$ is a gaining query word and $P(w_i)$ = 0.5 if $w_i$ is a declining query word. This reflects the fact that a gaining query is more important than a declining one. The function $f\left(R_{i,j}\right)$ takes a ranked number and converts into a weight for each word. A high ranked query is more important in this case.

We now integrate *GZ* as an additional score into Equation (1) in order to increase the probability of partition $p_j$:

$$Score_{gz}(d_i, p_j) = \sum_{w \in d_i} \left( P(w|p_j) \times \log \frac{P(w|p_j)}{P(w|C)} + \beta GZ(p_j, w) \right) \tag{6}$$

where $\beta$ is the weight for the *GZ* function which is obtained from an experiment and represented by a real number between 0 and 1.

## 6 Evaluation

Our proposed enhancements are evaluated by comparing their performance in determining the timestamp with experimental results from using the JRH approach as baseline. In this section, we will describe experimental setting, experiments and results.

### 6.1 Experimental Setting

In order to assign timestamp to a document, a reference corpus consisting of documents with known dates is required for comparison. A temporal language model is then created from the reference corpus. In fact, the temporal language model is intended to capture word usage within a certain time period. Two mandatory properties of the reference corpus are 1) it should consist of documents from various domains, and 2) it should cover the time period of a document to be dated.

We created a corpus collection from the Internet Archive [5] by downloading the history of of web pages, mostly web versions of newspapers (e.g., ABC News, CNN, New York Post, etc., in total 15 sources). The corpus collection covers on average 8 years for each source and the total number of web pages is about 9000 documents, i.e., the web pages in the corpus collection have on average been retrieved once every five day by the Internet Archive crawler.

### 6.2 Experiments

In order to evaluate the performance of the enhanced temporal language models, the documents in the corpus collection are partitioned into two sets ($C_{train}$, $C_{test}$). $C_{train}$ is used as a training set and to create a temporal language model. $C_{test}$ is used as a testing set and to estimate timestamps of documents (note that we actually have the correct timestamps of these documents so that the precision of estimation can be calculated).

The training set $C_{train}$ must meet the two properties mentioned above. This can be achieved by creating it based on news sources of various genres that cover the time period of documents to be dated. We choose 10 news sources from the corpus collection to build the training set. To create $C_{test}$, we randomly select 1000 documents from the remaining 5 news sources as a testing set.

In our experiments, we use two performance measures: precision and recall. Precision in our context means the fraction of processed documents that are correctly dated, while recall indicates the fraction of correctly dated documents that are processed. A recall lower than 100% is essentially the result of using confidence of timestamping to increase precision.

The experiments are conducted in order to study three aspects: 1) semantic-based preprocessing, 2) temporal entropy (*TE*) and Google Zeitgeist (*GZ*), and 3) confidence in the timestamp-estimation task. Unfortunately, we were unable to evaluate our proposed interpolation because of a too short time span (only 8 years) in the corpus collection. However, we use linear interpolation as proposed by Kraaij [7] in our experiments, and the smoothing parameter $\lambda$ is set to 0.1.

We evaluate the performance of the techniques repeating each experiment 10 times on different testing sets, which all are created based on random sampling. Averaged precision and recall are measured for each experiment.

**Experiment A:** In this experiment, we evaluate the performance of semantic-based preprocessing. The experiment is conducted on different combinations of semantic methods. In A.1, we study the effect of concept extraction. $C_{train}$ is created as a training language model with the preprocessing steps: POS tagging, WSD, concept extraction and word filtering. In A.2, we study the effect of collocation extraction. $C_{train}$ is created as a training language model with the preprocessing steps: POS tagging, collocation, WSD and word filtering. In A.3, $C_{train}$ is created as a training language model with the preprocessing steps: POS tagging, collocation extraction, WSD, concept extraction and word filtering. In all experiments, timestamp is determined for documents in $C_{test}$. Precision is measured for each combination of semantic-based techniques.

**Experiment B:** In order to evaluate the performance of temporal entropy and use of Google Zeitgeist statistics, we create a training language model on $C_{train}$ in two ways: using the semantic-based preprocessing in A.3 and without semantic-based preprocessing. For each document in $C_{test}$ the timestamp is determined using Equations (4) and (6). Precision is measured for each scoring technique.

**Experiment C:** Similar to a classification task, it is necessary to know how much confidence the system has in assigning a timestamp to a document. This can for example be used as feedback to a user, or as part of a subsequent query process where we want to retrieve documents from a particular time only of the confidence of the timestamp is over a certain threshold. Confidence is measured by the distance of scores of the first and the second ranked partitions and it is given as follows. $Conf(Time(d_i)) = \log \frac{Score(d_i,p_m)}{Score(d_i,p_n)}$ where $p_m$ and $p_n$ are the first two partitions that give the highest scores to a document $d_i$ computed by Equation (1). A language model is created for $C_{train}$ and, for each document in $C_{test}$, timestamp is determined by varying a confidence threshold. We measure precision and recall for each level of confidence.

### 6.3 Results

Fig. 4(a) (also presented in tabular form in Table 1) presents precision of results from determining timestamp for different granularities using the baseline technique (the JRH approach) and combinations of different preprocessing techniques (A.1/A.2/A.3). As can be seen, by adding semantic-based preprocessing higher precision can be obtained in almost all granularities except for 1-week (where only using concept extraction outperforms the baseline). The observation indicates that using a 1-week granularity, the frequency of a collocation in each week is not so different. For example, news related to "tsunami" were reported for about 6 weeks (during December 2004 and January 2005) and each week had almost the same frequency of collocations such as "tsunami victim" and "tsunami survivor". Thus the probability of a collocation is distributed in the case of a small granularity and it is hard to gain a high accuracy for any particular partition. On the other hand, as soon as the granularity becomes more coarse, usage of collocations are quite distinct, as can be seen from the results of 1-month, 3-month, 6-month and 12-month.

Fig. 4(b) (also presented in tabular form in Table 1) illustrates precision of results from determining timestamp when using temporal entropy (TE) without semantic-based preprocessing, Google Zeitgeist statistics without semantic-based preprocessing (GZ), temporal entropy with semantic-based preprocessing (S-TE), and Google Zeitgeist statistics with semantic-based preprocessing (S-GZ). As can be seen, without semantic-based preprocessing, TE only improves accuracy greatly in 12-month while in other granularities its results are not so different to those of the baseline, and GZ does not improve accuracy in all granularities. In contrast, by applying semantic-based preprocessing first, TE and GZ obtain high improvement compared to the baseline in almost all granularities except for 1-week which is too small granularity to gain high probabilities in distinguishing partitions.

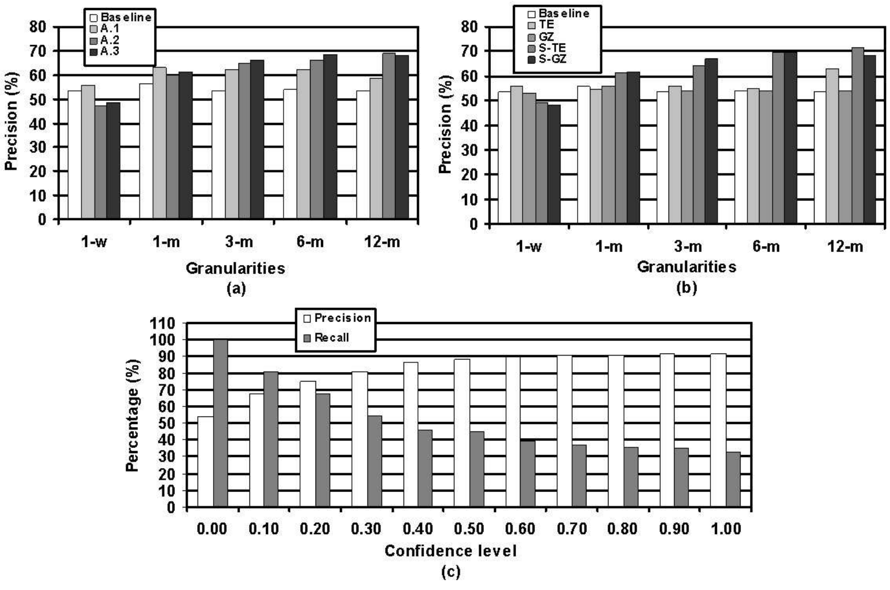


**Fig. 4.** Results from experiments A, B and C

From our observation, semantic-based preprocessing generates collocations as well as concept terms which are better in separating time partitions than single words. Those terms are weighted high by its temporal entropy. Similarly, most of the keywords in Google Zeitgeist statistics are noun phrases, thus collocations and concepts gains better GZ scores. This results in a high probability in determining timestamp.

Fig. 4(c) shows how the confidence level affects the accuracy of determining a timestamp. If the confidence level is 0, recall is 100% but precision is only 54.13%. On the other hand, if the confidence level is 1.0, precision is up to 91.35% but recall decreases to 33%. As shown in the figure, a high confidence threshold gives a high precision in determining the timestamp of documents, whereas a document with a correctly estimated date might be discarded. Thus the confidence level can be used to provide more reliable results.

## 7 Conclusion and Future Work

We have in this paper described several methods that increase the quality of determining timestamp of non-timestamped documents. Extensive experiments show that our approaches considerably increases quality compared to the baseline based on the previous approach by de Jong et al.

In order to increase reliability of timestamp-determination, we can take into account the confidence measure. In this way, applications that require high precision of results can choose to only use documents where the timestamp has been determined with high confidence.

There are several issues we intend to study as part of future research. First, our word interpolation method is an interesting idea in improving the language model. How-

| | Experiment A | | | | Experiment B | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Granularities | Baseline | A.1 | A.2 | A.3 | Baseline | TE | GZ | S-TE | S-GZ |
| 1-w | 53.430 | 55.873 | 47.072 | 48.365 | 53.430 | 55.725 | 53.050 | 49.126 | 48.423 |
| 1-m | 56.066 | 62.873 | 59.728 | 61.152 | 56.066 | 54.629 | 56.026 | 61.196 | 61.540 |
| 3-m | 53.470 | 62.076 | 65.069 | 66.360 | 53.470 | 55.751 | 54.030 | 64.525 | 67.008 |
| 6-m | 53.971 | 62.051 | 66.065 | 68.712 | 53.971 | 54.797 | 54.271 | 69.605 | 69.824 |
| 12-m | 53.620 | 58.307 | 69.005 | 68.216 | 53.620 | 63.104 | 53.947 | 71.564 | 68.366 |

**Table 1.** Precision in experiments A and B

ever, not every word should be interpolated in the same manner, thus we could apply a weighting scheme to words and interpolate only significant words.

## References

1. O. Alonso and M. Gertz. Clustering of search results using temporal attributes. In *Proceeding of the 29th SIGIR*, 2006.
2. K. Berberich, S. J. Bedathur, T. Neumann, and G. Weikum. A time machine for text search. In *Proceedings of SIGIR'2007*, 2007.
3. F. de Jong, H. Rode, and D. Hiemstra. Temporal language models for the disclosure of historical text. In *Proceedings of AHC'2005 (History and Computing)*, 2005.
4. Google Zeitgeist. `http://www.google.com/press/zeitgeist.html`.
5. Internet Archive. `http://archive.org/`.
6. A. Klose, A. Nfirnberger, R. Kruse, G. Hartmann, and M. Richards. Interactive text retrieval based on document similarities.
7. W. Kraaij. Variations on language modeling for information retrieval. *SIGIR Forum*, 39(1):61, 2005.
8. X. Li and W. B. Croft. Time-based language models. In *Proceedings of CIKM'2003*, 2003.
9. 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.
10. 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.
11. 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.
12. C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 999.
13. K. Nørvåg. The design, implementation, and performance of the V2 temporal document database system. *Journal of Information and Software Technology*, 46(9):557–574, 2004.
14. K. Nørvåg. Supporting temporal text-containment queries in temporal document databases. *Journal of Data & Knowledge Engineering*, 49(1):105–125, 2004.
15. M. G. Omar Alonso and R. Baeza-Yates. On the value of temporal information in information retrieval. *ACM SIGIR Forum*, 41(2):35–41, 2007.
16. J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR'1998*, 1998.
17. R. Swan and J. Allan. Extracting significant time varying features from text. In *Proceedings of CIKM'1999*, 1999.
18. R. Swan and D. Jensen. Timemines: Constructing timelines with statistical models of word usage. In *Proceedings of KDD-2000 Workshop on Text Mining*, 2000.
19. C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, 2004.