TASE: A Temporal Aware Search Engine

Sheng Lin, Peiquan Jin, Lihua Yue
School of Computer Science and Technology
University of Science and Technology of China
Hefei, China, 230027
linsh@mail.ustc.edu.cn, {jpq, llyue}@ustc.edu.cn

ABSTRACT
Most Web pages contain temporal information, which can be utilized by search engines to improve searching performance for users. However, traditional search engines have little support in processing temporal-textual Web search. Aiming at solving this problem, in this paper we present and implement a prototype system for temporal-sensitive queries called TASE (Temporal Aware Search Engine). TASE extracts the temporal expressions for each Web page and calculates the relevant score between the web page and each temporal expression, and then re-rank search results based on the temporal information of the documents and the queries. It is demonstrated that TASE is helpful to improve the effectiveness of temporal-textual Web queries.

Categories and Subject Descriptors
H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval.


Keywords: Web search, time, Re-ranking.

1. INTRODUCTION
Temporal information plays an important role in many research areas such as information extraction, topic detection, question answering, query log analysis, and Web search. Temporal information usually appears in Web pages as temporal expressions, which are typically divided into two types, namely explicit expressions, e.g., March 7, 2012, and implicit expressions, e.g., Today. Recent research on Web search shows that a lot of Web queries contain temporal keywords, either implicit temporal words [1] or explicit temporal expressions [2].

Traditional search engines such as Google and Bing are mainly focusing on textual search. They have very limited consideration on the temporal information contained in Web pages, i.e., they treat temporal information as textual keywords, or only support publication date based search. However, those ways both have some critical problems in dealing with time-related Web search. For example, the search engine cannot find the relevance between a query temporal expression “1996” and a document which mentions “1996”.

Aiming at solving this problem, in this paper we present and implement a prototype system for temporal-sensitive queries called TASE (Temporal Aware Search Engine). The major features of TASE can be described as follows:

1) TASE extracts the temporal expressions for each Web page and calculates the relevant score between the web page and each temporal expression. In comparison with traditional approach, it distinguishes the temporal expressions with their relevant score and takes the containment relationship among the temporal expressions into consideration.

2) TASE combines the temporal similarity and the textual similarity to re-rank the search results, and it can improve the effectiveness obviously for temporal-sensitive queries.

2. TASE ARCHITECTURE

Figure 1 shows the architecture of our system. TASE can be divided into four major modules as follows:

1) Candidate documents extraction module. It extracts the original Top-K documents from the search results which are used as the candidate documents.

2) Temporal expressions extraction module. It extracts all the temporal expressions in each candidate document, including the explicit temporal expressions and the implicit temporal expressions. It will be discussed in Section 3.1.

3) Relevant score calculation module. The relevant score between a temporal expression and a Web page will be...
calculated in this module. We will represent the score model in our algorithm in details later in Section 3.2.

(4) Temporal similarity calculation module. It calculates the similarity between the temporal expressions in a query and a document, and it will be described in Section 3.3.

(5) Re-ranking module. In this module, it used the temporal similarity and the original textual similarity to determine the final relevant score of a document. Section 3.4 will give more details about it.

3. KEY TECHNOLOGIES

3.1 Temporal Expressions Extraction

The explicit temporal expressions can be recognized by many time annotation tools, such as GUTime, which is part of the TARSQI (Temporal Awareness and Reasoning Systems for Question Interpretation) toolkit (TTK) [3], and they get high accuracy ratio in extraction. In this paper, we employ the GUTime tool to extract explicit temporal expressions.

The biggest difference of recognition between the explicit and implicit temporal expressions is that the implicit temporal expressions need to determine a reference time, so choosing the right reference is the key to the identification of the implicit temporal expression. Zhao [4] proposed a novel reference time dynamic-choosing mechanism which considers the global reference time and local reference time respectively.

In this paper, we classify temporal expressions two classes. One is called Global Time (GT) whose temporal semantics is independent of the local context, and takes the report time or publication time as the referent. Another one, Local Time (LT), makes reference to the narrative time in text above on account of depending on the current context. Table 1 gives some examples of GT and LT in real texts.

Table 1. Common Global Temporal Expressions and Local Temporal Expressions

<table>
<thead>
<tr>
<th>Class</th>
<th>Sub-class</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>GT</td>
<td>year</td>
<td>last year</td>
</tr>
<tr>
<td></td>
<td>month</td>
<td>next month</td>
</tr>
<tr>
<td></td>
<td>day</td>
<td>this Friday</td>
</tr>
<tr>
<td>LT</td>
<td>year</td>
<td>that year</td>
</tr>
<tr>
<td></td>
<td>month</td>
<td>October</td>
</tr>
<tr>
<td></td>
<td>day</td>
<td>the second day</td>
</tr>
</tbody>
</table>

In our approach, there is a reference time table which is used to hold full reference time for the whole text, and we need to update and maintain it dynamically after each normalizing process. The time table consists of two parts: Global Reference Time (GRT) and Local Reference Time (LRT).

In Figure 2, we notice that different classes of time dynamically and automatically choose references based on their respective classes rather than doing it using the fixed value or the inconsiderate rule under the static mechanism.

3.2 Relevant Score Calculation

In this section, we consider two aspects when calculating the relevant score of a temporal expression, namely the term frequency of the temporal expression and the relevance between temporal expressions. Here, we define the score of a temporal expression as a sum of an explicit score and an implicit score. The explicit score is related to the term frequency of a temporal expression, and accordingly the implicit score is related to the contribution made by all its children expressions. Table 2 shows the parent-child relationships among all the six time granularities we consider.

Table 2. The parent-child temporal relationships

<table>
<thead>
<tr>
<th>time granularity</th>
<th>parent time granularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAY</td>
<td>MONTH</td>
</tr>
<tr>
<td>MONTH</td>
<td>QUARTER</td>
</tr>
<tr>
<td>QUARTER</td>
<td>HALF</td>
</tr>
<tr>
<td>HALF</td>
<td>YEAR</td>
</tr>
<tr>
<td>YEAR</td>
<td>DECADE</td>
</tr>
</tbody>
</table>

The explicit score $ES(T_i)$ is defined as the term frequency of $T_i$ in the article. As compared to implicit temporal expressions $TF_{ITE}(T_i)$, the explicit temporal expressions $(TF_{ITE}(T_i))$ are more accurate in the extraction. So we add a weighting factor $d$ to the implicit temporal expressions. The explicit score of $T_i$ is defined as formula (1).

$$ES(T_i) = TF_{ITE}(T_i) + d \cdot TF_{ITE}(T_i).$$  \quad (1)

Here, $TF_{ITE}(T_i)$ refers to the term frequency of the explicit temporal expressions which are recognized as $T_i$, $TF_{ITE}(T_i)$ refers to the term frequency of the implicit temporal expressions which are calculated as $T_i$, $d$ is the weighting factor.

The implicit score $IS(T_i)$ is related to all the scores of its children, we denoted as $C_1, C_2, \ldots, C_n$, respectively, and we use the letter N to represent how many children unit $T_i$ contains. For example, if the granularity of $T_i$ is MONTH, then the value of N is 30 because a month contains about 30 days. Here, we use the factor $a$ to represent how much contribution the children of $T_i$ make. So the implicit score $IS(T_i)$ can be defined as formula (2).

$$IS(T_i) = TF_{ITE}(T_i) + d \cdot TF_{ITE}(T_i).$$  \quad (2)
\[ IS(T_1) = \frac{1}{\alpha \times N} \sum_{i=1}^{n} S(C_i) \]  

(2)

### 3.3 Temporal Similarity Calculation

Berberich, K. et al. [5] propose a method to calculate the temporal similarity. Based on it, we use formula (3) to calculate the temporal relevant score. It additionally takes the relevant score defined in Section 3.2 into consideration. So it can emphasize the temporal expressions with high relevant score.

\[ P(t_q | t_d) = \frac{\text{Score}(t_d) \cdot |t_q \cap t_d|}{|t_q| \cdot |t_d|} \]  

(3)

In this formula, \( t_q \) is one of the temporal expressions \( q_{\text{time}} \) in a query. \( t_d \) is one of the temporal expressions \( d_{\text{time}} \) in a document. \( \text{Score}(t_d) \) is the relevant score of \( t_d \). Berberich, K. also provides an efficient way to compute \( |t_q \cap t_d|, |t_q| \) and \( |t_d| \) in his temporal expression model. To take all the temporal expressions in queries and documents into account, the temporal similarity can be defined as formula (4).

\[ S''(q_{\text{time}}, d_{\text{time}}) = \prod_{t_q \in \text{time}} \left( \frac{1}{|t_q|} \sum_{t_d \in \text{time}} P(t_q | t_d) \right) \]  

(4)

### 3.4 Re-ranking Algorithm

One of the previous time-aware ranking methods is integrating the temporal similarity into existing language model, e.g. [5], [6]. To integrate to the probabilistic relevance model and vector space model, we can use a mixture model linearly combining textual and temporal similarity, e.g. [7], [8]. We Re-rank the documents by a linear combination of the original textual similarity score \( S'(q_{\text{word}}, d_{\text{word}}) \) and the temporal similarity score \( S''(q_{\text{time}}, d_{\text{time}}) \). Both similarity scores are needed to be normalized. For example, in this paper, all the similarity scores are divided by the maximum score of the Top-K documents. Given a temporal query \( q \), a document \( d \) will be ranked according to a score computed as formula (5). The parameter \( \alpha \) indicates the importance of textual similarity and temporal similarity.

\[ S(q,d) = (1 - \alpha) \cdot S'(q_{\text{word}}, d_{\text{word}}) + \alpha \cdot S''(q_{\text{time}}, d_{\text{time}}) \]  

(5)

### 4. DEMONSTRATION

The corpus we use for demonstration consists of 1,812,933 English news articles crawled from the New York Times website. It contains a total of 6,455,985 temporal expressions, and it includes 3,763,923 (58%) explicit temporal expressions and 2,692,062 (42%) implicit temporal expressions. More specifically, it mainly contains 3,157,156 DAY expressions (49%), 2,317,796 YEAR expressions (36%) and 959,747 MONTH expressions (15%).

All the documents are indexed and retrieved using the Apache Lucene version 3.5.0. There are two modes for retrieval [1]: 1) inclusive and 2) exclusive. For inclusive, both query terms and a temporal expression comprise a query \( q_{\text{incl}} \). For exclusive, only query terms constitute \( q_{\text{excl}} \), and a temporal expression is excluded from \( q_{\text{excl}} \). The Lucene’s default weighting function is regarded as the textual similarity.

We use a Web interface to demonstrate our system. Figure 3 is an example of demonstration. For more details, we provide a video to show the demonstration process, which can be found in http://home.ustc.edu.cn/~linsh/tase_demo/index.htm.

![Figure 3. TASE Web interface](image_url)

### 5. CONCLUSIONS

Aiming to solve the limited consideration on temporal information of traditional search engine, we build a prototype system named TASE (Temporal Aware Search Engine). In this paper, we describe the architecture and key technologies of our system. We will do more experiments to verify our algorithm in future.

### 6. REFERENCES


