Extending an Information Retrieval System through Time Event Extraction

Pierpaolo Basile, Annalina Caputo, Giovanni Semeraro, and Lucia Siciliani

Department of Computer Science - University of Bari Aldo Moro
Via E. Orabona, 4 - 70125 Bari (ITALY)
e-mail: {pierpaolo.basile@uniba.it, annalina.caputo@uniba.it,
giovanni.semeraro@uniba.it, siciliani.lu@gmail.com}

Abstract. In this paper we propose an innovative Information Retrieval system able to manage temporal information. The system allows temporal constraints in a classical keyword-based search. Information about temporal events is automatically extracted from text at indexing time and stored in an ad-hoc data structure exploited by the retrieval module for searching relevant documents. Our system can search textual information that refers to specific period of times. We perform an exploratory case study indexing all Italian Wikipedia articles.

1 Introduction

Identifying specific pieces of information related to a particular time period is a key task for searching past events. Although this task seems to be marginal for Web users [18], many search domains, like enterprise search, or lately developed information access tasks, such as Question Answering [20] and Entity Search, would benefit from techniques able to handle temporal information.

The capability of extracting and representing temporal events mentioned in a text can enable the retrieval of documents relevant for a given topic pertaining to a specific time. Nonetheless, the notion of *temporal* in the retrieval context has often being associated with the dynamic dimension of a piece of information, i.e. how it changes over time, in order to promote freshness in results. Such kind of approaches focus on when the document was published (*timestamp*) rather than the temporal event mentioned in its content (*focus time*). While traditional search engines take into account temporal information related to a document as a whole, our search engine aims to extract and index single events occurring in the texts, and to enable the retrieval of topics related to specific temporal events mentioned in the documents. In particular, we are interested in retrieving documents that are relevant for the user query, and also match some temporal constraints. For example, the user could be interested in a particular topic —strumenti musicali (*musical instrument*)— related to a specific time period —inventati tra il 1300 ed il 1500 (*invented between 1300 and 1500*)—.

However, looking for happenings in a specific time span requires further, and more advanced, techniques able to treat temporal information. Therefore, our goal is to merge features of both Information Retrieval (IRS) and Temporal Extraction Systems (TES). While an IRS allows us to handle and access the information included in texts, TES locate temporal expressions. We define this kind of system "Time-Aware IR" (TAIR).

In the past, several attempts have been made to exploit temporal information in IR systems [2], with an up-to-date literature review and categorization provided in [7]. Most of these approaches exploit time information related to the document in order to improve the ranking (recent documents are more relevant) [9], cluster documents using temporal attributes [1,3], or exploit temporal information for effectively present documents to the user [16]. However, just a handful of work have focused on temporal queries, that is the capability of querying a collection with both free text and temporal expression [4]. Alonso et al. pointed out as this kind of tasks needs the combination of results from both the traditional keyword-based and the temporal retrieval that can give rise to two different result sets. Vandenbussche and Teissèdre [23] dealt with temporal search in the context of both the Web of Content and the Web of Data, but differently from our system, they relied on an ontology of time for temporal queries [11]. Kanhabua and Nørvåg [13] defined semantic- and temporal-based features for a learning to rank approach by extracting named entities and temporal events from the text. Similarly to our approach, Arikan et al. [5] considered the query as composed by a keyword and a temporal part. Then, the two queries were addressed by computing two different language model-based weights. Exploiting a similar model, Berberich et al. [6] developed a framework for dealing with uncertainty in temporal queries. However, both approaches drawn the probability of the temporal query out of the whole document, thus neglecting the pertinence of temporal events at a sentence level. In order to overcome such a limitation, Matthews et al. [17] introduced two different types of indexes, at a document and a sentence level, with the latter associated with content date.

Preliminary to indexing and retrieval, the Information Extraction phase aims to extract temporal information, and its associated events, from text. In this area [15], several approaches aim at building structured knowledge sources of temporal events. In [12] the authors describe an extension of the YAGO knowledge base, in which entities, facts, and events are anchored in both time and space. Other work exploit Wikipedia to extract temporal events, such as those reported in [10, 14, 25]. Temporal extraction systems can locate temporal expressions and normalize them making this information available for further processing. Currently, there are different tools that can make this kind of analysis on documents, like SUTime [8] or HeidelTime [21] and other systems which took part in TempEval evaluation campaigns. Temporal extraction is not the main focus of this paper, then we remand the interested reader to the TempEval description task papers [22,24] for a wider overview of the latest state-of-the-art temporal extraction systems.

The paper is organized as follows: Section 2 provides details about the model behind our TAIR system, while Section 3 describes the implementation of our model. Section 4 reports some use cases of the TAIR system which show the potential of our approach, while Section 5 closes the paper.

2 Time-Aware IR Model

A TAIR model should be able to tackle some problems that emerge from temporal search [23], that is: 1) the extraction and normalization of temporal references, 2) the representation of the temporal expressions associated to documents, and 3) the ranking under the constraint of keyword- and temporal-queries.

Our TAIR model consists of three main components responsible to deal with these issues, as sketched in Figure 1:

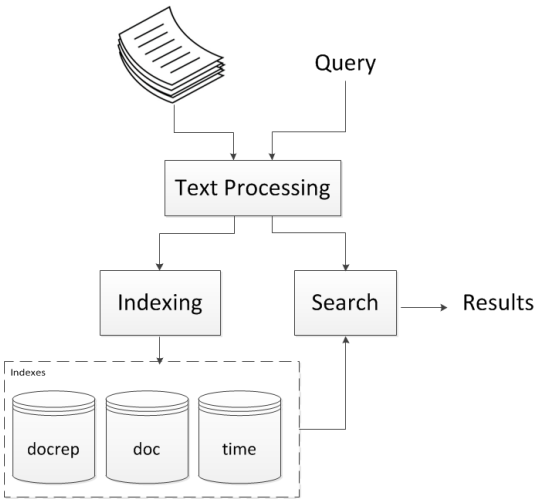


Fig. 1: The IR time-aware Model

Text processing It automatically extracts time expressions from text. The extracted expressions are normalized in a standard format and sent to the indexing component;

Indexing This component is dedicated to index both textual and temporal information. During the indexing, text fragments are linked to time expressions. The idea behind this approach is that the context of a temporal expression is relevant;

Search It analyzes the user query composed by both keywords and temporal constraints, and performs the search over the index in order to retrieve relevant information.

2.1 Text Processing Component

Given a document as input, the text processing component provides as output the *normalized* temporal expressions extracted from the text, along with information about positions in which the temporal expressions are found. For this purpose we adopt a standard annotation language for temporal expressions called TimeML [19]. We are interested in expressions tagged with the TIMEX3 tag that is used to mark up explicit temporal expressions, such as times, dates and durations. In TIMEX3 the value of the temporal expression is normalized according to 2002 TIDES guideline, an extension of the ISO-8601 standard, and is stored in an attribute called *value*. An example of TIMEX3 annotation for the sentence "before the 23th May 1980" is reported below:

```
<TimeML>
    before the
    <TIMEX3 tid="t3" type="DATE" value="1980-05-23">
        23th May 1980
    </TIMEX3>
</TimeML>
```

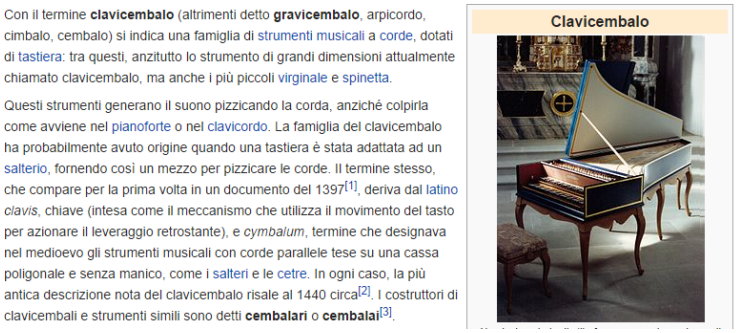
Where `tid` is a unique identifier, `type` can assume one of the types between: `DATE`, `TIME`, `DURATION`, and `SET`, while the `value` attribute contains the temporal information that varies accordingly to the type.
ISO-8601 normalizes temporal expressions in several formats. For example, "May 1980" is normalized as "1980-05", while "23th May 1980" as "1980-05-23". We choose to normalize all dates using the pattern *yyyy-mm-dd*. All temporal expressions not compliant to the pattern, such as "1980", must be normalized retaining the lexicographic order between dates. Our solution consists in normalizing all temporal expressions in the form of *yyyy* or *yyyy-mm* to the last day of the previous year or month, respectively. In our previous example, the expression "1980" is normalized as 19791231. Similarly, the expression "1980-05" is normalized as "1980-04-30". Moreover, the text processing component applies several normalization rules to correctly identify seasons, for example the TimeML tag for Spring "yyyy-SP" is normalized as "yyyy-03-20".
Using the correct normalization, the order between periods is respected. In conclusion the text processing component extracts temporal information and correctly normalized them to make different time periods comparable.

2.2 The Indexing Component

After the text processing step, we need to store and index data. In our model we propose to store both documents and temporal expressions in three separated data indexes, as reported in Figure 1.
The first index ($docrep$) stores the text of each document (without processing) with an id, a numeric value that unequivocally identifies the document. This index is used to store the document content only for the presentation purpose. The second index (doc) is a traditional inverted index in which the text of each document is indexed and used for keyword-based search. Finally, the last index ($time$) stores temporal expressions found in each document. For each temporal expression, we store the following information:

- The document id;
- The normalized value of the time expression according to the normalization procedure described in Section 2.1;
- The start and end offset of the expression in the document, useful for highlighting;
- The context of the expression: the context is defined by taking all the words that can be found within n characters before and after the time expression. The context is indexed and used by the search component during the retrieval step. The idea is to keep trace of the context where the time expression occurred. The context is tokenized and indexed and exploited in conjunction with the keyword-based search, as we explained in Section 2.3.

It is important to note that a document could have many temporal expressions, for each of these an entry in the $time$ index is created. For example, given the



Clavicembalo

Da Wikipedia, l'enciclopedia libera.

Con il termine **clavicembalo** (altrimenti detto **gravicembalo**, arpicordo, cimbalo, cembalo) si indica una famiglia di strumenti musicali a corde, dotati di tastiera: tra questi, anzitutto lo strumento di grandi dimensioni attualmente chiamato clavicembalo, ma anche i più piccoli virginale e spinetta.

Questi strumenti generano il suono pizzicando la corda, anziché colpirla come avviene nel pianoforte o nel clavicordo. La famiglia del clavicembalo ha probabilmente avuto origine quando una tastiera è stata adattata ad un salterio, fornendo così un mezzo per pizzicare le corde. Il termine stesso, che compare per la prima volta in un documento del 1397[1], deriva dal latino *clavis*, chiave (intesa come il meccanismo che utilizza il movimento del tasto per azionare il leveraggio retrostante), e *cymbalum*, termine che designava nel medioevo gli strumenti musicali con corde parallele tese su una cassa poligonale e senza manico, come i salteri e le cetre. In ogni caso, la più antica descrizione nota del clavicembalo risale al 1440 circa[2]. I costruttori di clavicembali e strumenti simili sono detti **cembalari** o **cembalai**[3].

Clavicembalo

Fig. 2: Wikipedia page example.

Wikipedia page in Figure 2, we store its whole content as reported in Table 1a, while we tokenize and index the page as shown in Table 1b. The most interesting part of the indexing step is the storage of temporal expressions. As depicted in Table 1c, for each temporal expression we store the normalized time value, in this case "13961231", and the start and end offset of the expression in the text. Finally, we tokenize and index the context in which the expression occurs. In Table 1c, in italics is reported the left context, while the right context is reported in bold. Examples are reported according to the Italian version of Wikipedia, but the indexing step is language independent.

2.3 The Search Component

The search component retrieves relevant documents according to the user query q containing temporal constraints. For this reason we need to make temporal expressions in the query compliant with the expressions stored in the index. The query is processed by the Text Component in order to extract and normalize the time expressions.

The query q is represented by two parts: q_k contains keywords, while q_t only the normalized time expressions. q_k is used to retrieve from the doc index a first results set RS_{doc}. Thus, both q_k and q_t are used to query the $time$ index producing the results set RS_{time}. The search in $time$ index is limited to those documents belonging to RS_{doc}. In RS_{time}, text fragments have to match the time constraints expressed in q_t, while the matching with the keyword-based query q_k is optional. The optional matching with q_k has the effect of promoting those contexts that satisfy both the temporal constraints and the query topics, while not completely removing poorly matching results. The motivation behind this approach is twofold: through RS_{doc} we retrieve those documents relevant for the query topic, while RS_{time} contains the text fragments that match the time query q_t and are related to the query topic.

For example given the query q ="clavicembalo [1300 TO 1400]", we identify the two fields: q_k ="clavicembalo" and q_t = [12991231 TO 13991231]. It is

Field	*Value*
ID	42
Content	Con il termine clavicembalo (altrimenti detto gravicembalo, arpicordo, cimbalo, cembalo) si indica una famiglia di strumenti musicali a corde [...]

(a) *docrep* index.

Field	*Value*
ID	42
Content	{'Con', 'il', 'termine', 'clavicembalo', 'altrimenti', 'detto', 'gravicembalo', 'arpicordo', 'cimbalo', 'cembalo', 'si', 'indica', 'una', 'famiglia', 'di', 'strumenti', 'musicali', 'a', 'corde' [...] }

(b) *doc* index.

Field	*Value*
ID	42
Time	13961231
Start Offset	350
End Offset	354
Context	{'*Il*', '*termine*', '*stesso*', '*che*', '*compare*', '*per*', '*la*', '*prima*', '*volta*', '*in*', '*un*', '*documento*', '*del*', '**deriva**', '**dal**', '**latino**', '**clavis**', '**chiave**' [...] }

(c) *time* index.

Table 1: The three indices used by the system.

important to underline that in this example we adopted a particular syntax to identify range queries, more details about the system implementation are reported in Section 3.

The retrieval step produces two results sets: RS_{doc} and RS_{time}. Considering the query q in the previous example: RS_{doc} contains the doc 42 with a relevance score s_{doc}. While the results set RS_{time} contains the temporal expression reported in Table 1c with a score s_{time}. The last step is to combine the two results sets. The idea is to promote text fragments in RS_{time} that comes from documents that belong to RS_{doc}. We simply boost the score of each result in RS_{time} multiplying its score by the score assigned to its origin document in RS_{doc}. In our example the temporal expression occurring in RS_{time} obtains a final score computed as: $s_{doc} \times s_{time}$. We have chosen to boost score rather than linearly combine them, in this way we avoid the use of combination parameters.

Finally, we sort the re-ranked RS_{time} and provide it to the user as final result of the search. It is important to underline that our system does not produce a list of document as a classical search engine does, but we provide all the text passages that are both relevant for the query and compliant to temporal constraints.

3 System Implementation

We implemented our TAIR model in a freely available system[1] as an open-source software under the GNU license V.3. The system is developed in JAVA and extends the indexing and search open-source API Apache Lucene[2].
The text processing component is based on the HeidelTime tool[3] [21] to extract temporal information. We adopt this tool for two reasons: 1) it obtained good performance in the TempEval-3 task, and 2) it is able to analyze text written in several languages including the Italian. HeidelTime is a rule based system that can be extended to support other languages or specific domains.
Our system provides all the expected functionalities: text analysis, indexing and search. The query language supports all operators provided by the Lucene query syntax[4]. Moreover the temporal query q_t can be formulated using natural time expressions, for example "12 May 2014" or "yesterday". The search component tries to automatically translate the user query in the proper time expressions. However, the user can directly formulate q_t using normalized time expressions and query operators. Table 2 shows some time operators.

Query	*Description*
20020101	match exactly 1st January 2002
[20020101 TO 20030101]	match from 1st January 2002 to 1st January 2003
[∗ TO 20030101]	before 1st January 2003
[20020101 TO ∗]	after 1st January 2002
01??2002	any first day of the month in 2002, * should be used for multiple character match, for example 01*2002
20020101 AND 20020131	the first and last day of January 2002, AND and OR operator can be used to combine exact match and range query

Table 2: Example of time query operators.

Currently the system does not provide a GUI for searching and visualizing the results, but it is designed as an API. As future works we plan to extend the API with REST Web functionalities.

4 Use case

We decided to set up a case study to show the potentialities of the proposed IR framework. The case study involves the indexing of a large collection of docu-

[1] https://github.com/pippokill/TAIR
[2] http://lucene.apache.org/
[3] https://code.google.com/p/heideltime/
[4] http://lucene.apache.org/core/4_8_1/queryparser/org/apache/lucene/queryparser/classic/package-summary.html

ments and a set of example queries exploiting specific scenarios in which temporal expressions play a key role. Moreover, another goal is to provide performance information about the system in terms of indexing and query time, and index space.

We propose an exploratory use case indexing all Italian Wikipedia articles. Our choice is based on the fact that Wikipedia is freely available and contains millions of documents with many temporal events. We need to set some parameters: we index only documents with at least 4,000 characters, remove special pages (e.g. category pages), we set the context size in temporal index to 256 characters.

We perform the experiment on a virtual machine with four virtual cores and 32GB of RAM. Table 3 reports some statistics related to the indexing step. The indexing time is very high due to the complexity of the temporal extraction algorithm and the huge number of documents. We speed up the temporal event extraction implementing a multi threads architecture, in particular in this evaluation we enable four threads for the extraction.

$Statistics$	$Value$
Number of documents	168,845
Number of temporal expressions	6,615,430
Indexing time	68 hours
Indexing time (doc./min.)	41,38

Table 3: Indexing performance.

One of the most appropriate scenarios consists in finding events that happened in a specific date. For example, one query could be interested in listing all events happened on 29 April 1981. In this case the time query is "19810429" while the keyword query is empty. The first three results are shown in Table 4.

We report in bold the temporal expressions that match the query. It is important to note that in the first result the year "1981" appears distant from both the month and the day, but the Text Processing component is able to correctly recognize and normalize the date.

Another interesting scenario is to find events related to a specific topic in a particular time period. For example, Table 5 reports the first three results for the query: "terremoti tra il 1600 ed il 1700" (*earthquakes between 1600 and 1700*). This query is split in its keyword q_k ="terremoti" (*earthquakes*) and temporal component $q_t = [15991231 \text{ TO } 16991231]$.

Table 6 shows the usage of time query operators, in particular of wild-cards. We are interested in facts related to *computers* which happened in January 1984 using the time query pattern "198401??".

As reported in Table 6, the first two results regard events whose time interval encompasses the time expressed in the query, since they took place in 1984, while the third result shows an event that completely fulfil the time requirements expressed in the temporal query.

Result Rank	Wikipedia page	Time Context
1	Paul Breitner	nel **1981**, richiamato da Jupp Derwall, nel frattempo divenuto nuovo commissario tecnico della Germania Ovest, e con il quale aveva comunque avuto accese discussioni a distanza. Il "nuovo debutto" avviene ad Amburgo il **29 aprile** contro l'Austria.
2	...E tu vivrai nel terrore! L'aldilà	Warbeck e Catriona McColl, presente nei contenuti speciali del DVD edito dalla NoShame. Accoglienza. Il film uscì in Italia il **29 aprile 1981** e incassò in totale 747.615.662 lire. Distribuito per i mercati esteri dalla VIP International, ottenne un ottimo successo
3	RCS Media Group	L'operazione venne perfezionata il **29 aprile 1981**. Quel giorno una società dell'Ambrosiano (quindi di Calvi), la "Centrale Finanziaria S.p.A." effettuò l'acquisto del 40% di azioni Rizzoli

Table 4: Results for the query "19810429"

Result Rank	Wikipedia page	Time Context
1	Terremoto della Calabria dell'8 giugno 1638	Il terremoto dell'**8 giugno 1638** fu un disastroso terremoto che colpì la Calabria, in particolare il Crotonese e parte del territorio già colpito nei giorni **27 e 28 marzo del 1638**
2	Eruzione dell'Etna del 1669	**1669 10 marzo** - M = 4.8 Nicolosi Terremoto con effetti distruttivi nel catanese in particolare a Nicolosi in seguito all'eruzione dell'Etna conosciuta come Eruzione dell'Etna del **1669**. Il **25 febbraio** e l'**8 e 10 marzo del 1669** una serie di violenti terremoti.
3	Terremoto del Val di Noto del 1693	l'evento catastrofico di maggiori dimensioni che abbia colpito la Sicilia orientale in tempi storici.Il terremoto del **9 Gennaio 1693**

Table 5: Results for the query "earthquakes between 1600 and 1700"

Result Rank	Wikipedia page	Time Context
1	Apple III	L'Apple III, detto anche Apple ///, fu un personal computer prodotto e commercializzato da Apple Computer dal 1980 al **1984** come successore dell'Apple II
2	Home computer	Apple Macintosh **(1984)**, il primo home/personal computer basato su una interfaccia grafica, nonch il primo a 16/32-bit
3	Apple Macintosh	Apple Computer (oggi Apple Inc.). Commercializzato dal **24 gennaio 1984** al 1 ottobre 1985, il Macintosh il capostipite dell'omonima famiglia

Table 6: Results for the query "computer" with the temporal pattern "198401??"

5 Conclusions and Future Work

We proposed a "Time-Aware" IR system able to extract, index, and retrieve temporal information. The system expands a classical keyword-based search through temporal constraints. Temporal expressions, automatically extracted from documents, are indexed through a structure that enables both keyword- and time-matching. As a result, TAIR retrieves a list of text fragments that match the temporal constraints, and are relevant for the query topic. We proposed a preliminary case study indexing all the Italian Wikipedia and described some retrieval scenarios which would benefit from the proposed IR model.
As future work we plan to improve both recognition and normalization of time expressions, extending some particular TimeML specifications that in this preliminary work were not taken into account during the normalization process. Moreover, we will perform a deep "in-vitro" evaluation on a standard document collection.

Acknowledgements

This work fulfils the research objectives of the projects PON 01_00850 ASK-Health (Advanced System for the interpretation and sharing of knowledge in health care) and PON 02_00563_3470993 project "VINCENTE - A Virtual collective INtelligenCe ENvironment to develop sustainable Technology Entrepreneurship ecosystems" funded by the Italian Ministry of University and Research (MIUR).

References

1. Alonso, O., Gertz, M.: Clustering of Search Results Using Temporal Attributes. In: Proceedings of the 29th annual international ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 597–598. ACM (2006)

2. Alonso, O., Gertz, M., Baeza-Yates, R.: On the Value of Temporal Information in Information Retrieval. SIGIR Forum 41(2), 35–41 (2007)
3. Alonso, O., Gertz, M., Baeza-Yates, R.: Clustering and Exploring Search Results Using Timeline Constructions. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management. pp. 97–106. CIKM '09, ACM (2009)
4. Alonso, O., Strötgen, J., Baeza-Yates, R.A., Gertz, M.: Temporal Information Retrieval: Challenges and Opportunities. In: Proceedings of the 1st International Temporal Web Analytics Workshop (TWAW 2011). vol. 11, pp. 1–8 (2011)
5. Arikan, I., Bedathur, S.J., Berberich, K.: Time Will Tell: Leveraging Temporal Expressions in IR. In: Baeza-Yates, R.A., Boldi, P., Ribeiro-Neto, B.A., Cambazoglu, B.B. (eds.) Proceedings of the 2ND International Conference on Web Search and Web Data Mining, WSDM 2009, Barcelona, Spain, February 9-11, 2009. ACM (2009)
6. Berberich, K., Bedathur, S., Alonso, O., Weikum, G.: A Language Modeling Approach for Temporal Information Needs. In: Proceedings of the 32Nd European Conference on Advances in Information Retrieval. pp. 13–25. ECIR'2010, Springer-Verlag (2010)
7. Campos, R., Dias, G., Jorge, A.M., Jatowt, A.: Survey of Temporal Information Retrieval and Related Applications. ACM Computing Surveys 47(2), 15:1–15:41 (2014)
8. Chang, A.X., Manning, C.D.: SUTime: A library for recognizing and normalizing time expressions. In: LREC. pp. 3735–3740 (2012)
9. Elsas, J.L., Dumais, S.T.: Leveraging Temporal Dynamics of Document Content in Relevance Ranking. In: Proceedings of the 3rd ACM International Conference on Web Search and Data Mining. pp. 1–10. WSDM '10, ACM (2010)
10. Hienert, D., Luciano, F.: Extraction of Historical Events from Wikipedia. In: Proceedings of the First International Workshop on Knowledge Discovery and Data Mining Meets Linked Open Data. pp. 25–36 (2011)
11. Hobbs, J.R., Pan, F.: An Ontology of Time for the Semantic Web. ACM Transactions on Asian Language Information Processing (TALIP) - Special Issue on Temporal Information Processing 3(1), 66–85 (2004)
12. Hoffart, J., Suchanek, F.M., Berberich, K., Weikum, G.: YAGO2: A Spatially and Temporally Enhanced Knowledge Base from Wikipedia. Artificial Intelligence 194, 28–61 (2013)
13. Kanhabua, N., Nørvåg, K.: Learning to Rank Search Results for Time-sensitive Queries. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management. pp. 2463–2466. CIKM '12, ACM (2012)
14. Kuzey, E., Weikum, G.: Extraction of temporal facts and events from Wikipedia. In: Proceedings of the 2nd Temporal Web Analytics Workshop. pp. 25–32. ACM (2012)
15. Ling, X., Weld, D.S.: Temporal Information Extraction. In: Proceedings of the 24th Conference on Artificial Intelligence (AAAI 2010). Atlanta, GA. (2010)
16. Matthews, M., Tolchinsky, P., Blanco, R., Atserias, J., Mika, P., Zaragoza, H.: Searching through time in the New York Times. In: Proceedings of the Fourth Workshop on Human-Computer Interaction and Information Retrieval (HCIR 10). pp. 41–44 (2010)

17. Matthews, M., Tolchinsky, P., Blanco, R., Atserias, J., Mika, P., Zaragoza, H.: Searching through time in the New York Times. In: Proceedings of the 4th Workshop on Human-Computer Interaction and Information Retrieval, HCIR Challenge 2010. pp. 41–44 (2010)
18. Nunes, S., Ribeiro, C., David, G.: Use of temporal expressions in web search. In: Proceedings of the IR Research, 30th European Conference on Advances in Information Retrieval, pp. 580–584. ECIR'08, Springer-Verlag (2008)
19. Pustejovsky, J., Castano, J.M., Ingria, R., Sauri, R., Gaizauskas, R.J., Setzer, A., Katz, G., Radev, D.R.: TimeML: Robust Specification of Event and Temporal Expressions in Text. New Directions in Question Answering 3, 28–34 (2003)
20. Saurí, R., Knippen, R., Verhagen, M., Pustejovsky, J.: Evita: A Robust Event Recognizer for QA Systems. In: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing. pp. 700–707. ACL (2005)
21. Strötgen, J., Zell, J., Gertz, M.: HeidelTime: Tuning English and Developing Spanish Resources for TempEval-3. In: 2nd Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the 7th International Workshop on Semantic Evaluation. pp. 15–19. ACL (2013)
22. UzZaman, N., Llorens, H., Derczynski, L., Allen, J., Verhagen, M., Pustejovsky, J.: Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In: 2nd Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the 7th International Workshop on Semantic Evaluation. pp. 1–9. ACL (2013)
23. Vandenbussche, P.Y., Teissèdre, C.: Events Retrieval Using Enhanced Semantic Web Knowledge. In: Workshop DeRIVE 2011 (Detection, Representation, and Exploitation of Events in the Semantic Web) in cunjunction with 10th International Semantic Web Conference 2011 (ISWC 2011) (2011)
24. Verhagen, M., Sauri, R., Caselli, T., Pustejovsky, J.: SemEval-2010 Task 13: TempEval-2. In: Proceedings of the 5th International Workshop on Semantic Evaluation. pp. 57–62. ACL (July 2010)
25. Whiting, S., Jose, J., Alonso, O.: Wikipedia As a Time Machine. In: Proceedings of the Companion Publication of the 23rd International Conference on World Wide Web Companion. pp. 857–862. International World Wide Web Conferences Steering Committee (2014)