Recognizing Textual Entailment with Temporal Expressions in Natural Language Texts

Rui Wang
Building C 7.4
Saarland University
D-66123 Saarbrücken, Germany
rwang@coli.uni-sb.de

Yajing Zhang
DFKI GmbH
Stuhlsatzenhausweg 3
D-66123 Saarbrücken, Germany
yajing.zhang@dfki.de

Abstract

The TACTE system proposed in this paper focuses on one problem in natural language processing, namely recognizing textual entailment involving temporal expressions. The system consists of two components: one for temporal expression extraction and anchoring, and the other one for recognizing textual entailment based on events. The entailment rules are constructed using a small set of temporal expression relations and lexical resources. Several experiments are conducted, and various aspects of the system performance are illustrated. The evaluation on different data sets shows the great improvement of our TACTE system in comparison with the baseline. As a system potentially to be integrated into a larger framework, TACTE is shown to be very promising as a specialized module on entailment cases where temporal expression information is available.

1 Introduction

Textual inference has shown its importance in many natural language processing (NLP) areas, e.g. information extraction [16], question answering [11], etc. In recent years, recognizing textual entailment (RTE) challenges proposed by [3] aim to find a generic semantic framework for many NLP applications. This task is described as given a pair of text fragments, *Text* (**T**) and *Hypothesis* (**H**), the system is asked to tell whether the latter can be inferred (or entailed) by the former. In practise, it takes the human judgements as a gold standard, i.e. whether human beings consider **H** is true given that **T** is true.

RTE is quite challenging, since the state-of-the-art systems have an average accuracy of around 60% [3], which is only 10% more than a random guess. The main difficulties are: 1) various kinds of linguistic or even world knowledge need to be acquired and applied; and 2) too many cases of entailment are covered in the data sets of limited sizes. One reasonable way to solve the problem is to use the divide-and-conquer strategy, breaking down the problem and focusing on one part of it instead of solving it as a whole.

In this paper, we aim at entailment pairs which contain temporal expressions. On the one hand, we would like to benefit from the recent research on temporal expression extraction and anchoring; on the other hand, the current system can be further integrated into a larger framework as a specialized module for general textual inference systems.

The use of temporal expressions is based on the assumption that very often important clues to distinguish the main topic of a **T-H** pair and the subsidiary information are given by temporal information. However, temporal information about the temporal location of events is not always given explicitly by some date or time expression, but by relative references such as *the week before*. Therefore, a time anchoring component (TAC) is developed to resolve temporal expressions, construct a time line of events, and distinguish event information from other relating information.

Our approach has three main steps: 1) extracting and anchoring temporal expressions (cf. Sec. 3); 2) using temporal expressions as starting points to find corresponding events in the dependency structure (cf. Sec. 4); and 3) applying lexical resources and entailment rules between temporal expressions to detect the entailment relationship (cf. Sec. 5). For evaluation, we extract a subset of the RTE-2 and RTE-3 data sets and also semi-automatically constructed some additional data sets from TREC2003[1]. The promising experimental results (cf. Sec. 6) show the advantages of our system as well as the potential to be combined

[1]http://trec.nist.gov/pubs/trec12/t12_proceedings.html

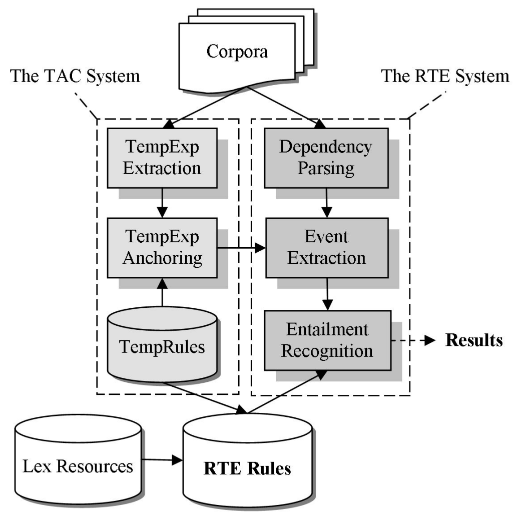

Figure 1. Architecture of the TACTE System.

with other methods.

2 Architecture

The TACTE system (cf. Figure 1) mainly consists of two components, RTE and TAC, and the entailment rules serve as the knowledge base to detect the entailment relation. The TAC system uses SProUT, an information extraction (IE) platform (cf. Sec. 3), to extract *Date* and *Time* expressions and anchors them based on manually defined rules. The RTE system pre-processes the texts using a dependency parser and later extracts the corresponding events based on the dependency structure using the temporal expressions as starting points. The entailment rules come from two sources: 1) lexical semantic resources and 2) entailment rules between temporal expressions. In the following sections, we will illustrate these components in detail.

3 Temporal Expression Anchoring

The core engine extracting temporal expressions in TAC is provided by SProUT [5], a multilingual platform developed for shallow natural language processing applications. SProUT combines finite state techniques with unification of typed feature structures (TFS). TFS provides a powerful device for representing and propagating information. Rules are expressed by regular expressions over input TFSs that get instantiated by the analysis. The uniform use of TFSs for input and output also allows for cascaded application of rule systems.

The representation of dates and times in TAC is based on *OWLTime* [8]. This ontology provides classes for representing temporal instants. The core date-time representation is the class *DateTimeDescription* that provides as properties fields for representing the day, month, year, hour, minute, second, weekday as well as the time zone. The use of *OWLTime* presupposes to some extent that dates or times are completely specified. But it poses some problems for the representation of partial and underspecified temporal expressions as used in natural language texts. The TAC component described here bridges the gap between temporal natural language expressions and *OWLTime* representations.

Both time points and durations are represented by the class *DateTimeInterval*[2] which references *DateTimeDescription* and *DurationDescription*. For better compliance with *TimeML*[3], *OWLTime* was extended by adding to the *DateTimeDescription* class properties for representing the week number (e.g. for representing the reference of expressions like *last week*), seasons (e.g. for references of *last summer*) and daytimes (e.g. *afternoon*) rather than representing these imprecise times directly as durations.

3.1 Two Types of Temporal Expression

In the temporal expression extraction process we distinguish two types of temporal expressions: time points and durations.

Time Points *DateTimeInterval* only specifies *DateTimeDescription* with following properties[4]: *day, month, year, hour, minute, second, pofd, dofw, weeknumber, pofm, pofy*. Among all these properties, the property *year* is obligatory which means each anchored time point must at least specify a value for *year*. Figure 2 shows the representation for the date *Friday October 24th, 1997*.

An important dimension to take into account for temporal resolution and computation is the *granularity* order of these properties. The order is similar to our intuition:

[*second* $<$ *minute* $<$ *hour* $<$ *pofd* $<$ *dofw* $<$ *day* $<$ *weeknumber* $<$ *pofm* $<$ *month* $<$ *pofy* $<$ *year*]

Durations *DateTimeInterval* can consist of *DateTimeDescription* or *DurationDescription*. *DurationDescription* contains properties of *days, months,*

[2]A time point described by a *DateTimeDescription* can be viewed as an interval according to its granularity or specificity, e.g. *yesterday* is an interval of the last 24 hours.

[3]http://www.timeml.org/site

[4]pofd: part-of-day, dofw: day-of-week, pofm: part-of-month, pofy: part-of-year

$$\begin{bmatrix} \textit{DateTimeInterval} & \\ \textsc{hasDateTimeDescription} & \begin{bmatrix} \textit{DateTimeDescription} & \\ \textsc{day} & \texttt{24} \\ \textsc{month} & \texttt{10} \\ \textsc{year} & \texttt{1997} \\ \textsc{dofw} & \texttt{5} \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \textit{DateTimeInterval} & \\ \textsc{hasBeginning} & \begin{bmatrix} \textit{DateTimeDescription} \\ \textsc{dofw}\ \ \texttt{2} \end{bmatrix} \\ \textsc{hasEnd} & \begin{bmatrix} \textit{DateTimeDescription} \\ \textsc{dofw}\ \ \texttt{4} \end{bmatrix} \\ \textsc{intervalBefore} & \#refDateInstance \end{bmatrix}$$

Figure 2. Representation of *Friday October 24th, 1997* **and** *from Tuesday to Thursday*

years, hours, minutes, seconds, weeks. Additionally thirteen relations defined in [1] describe the relation between *DurationDescription* and the reference time.

Due to the restricted granularity level of the reference time, a *DateTimeInterval* may have underspecified beginning and end points. Figure 2 shows the representation for *from Tuesday to Thursday*, where the reference time is *October 24th, 1997 (Friday).*

3.2 Anchoring of Temporal Expressions

To anchor temporal expressions, *Explicit* and *Relative expressions* are distinguished:

- *Explicit expressions* refer to a specific point or period of time. It can be unambiguously identified in a calendar, for instance, *June 6th, 2006.*
- *Relative expressions* refer to a specific point or period of time that can only be unambiguously identified with the help of a reference time given by context. Examples include *yesterday*, *two hours later*, *in summer*, etc.

Different from the division made by [6], we do not distinguish deictic and relative expressions, since both of them require a contextually given reference time to anchor the expression correctly. The difference is only in the type of context. We also consider duration expressions as intersecting with the explicit and deictic expressions. A time expression for duration can consist of explicit or relative expressions, for instance, *from June 6th, 2006 to June 9th, 2006*, *from today to tomorrow*, etc.

The reference time is context-dependent and dynamic. Currently when no explicit time is mentioned in the text or hypothesis, a default reference time is set to both. When another explicit time is mentioned in subsequent sentences, it can become the new reference time.

TAC also decides about the granularity level at which completion is necessary. The result inherits the granularity of the original incomplete expression. For instance, let the reference date be *October 24th, 1997 (Friday).* In Example (1) the granularity of the original expression *last Wednesday* is *dofw* and is anchored to *Wednesday October 15, 1997*, while Example (2) has the granularity of *minute* and will be anchored to *October 24th, 1997, 15:08.*

(1) The defence secretary William Cohen announced plans on *last Thursday.*

(2) The earthquake shook the province of Mindanao at *3:08 p.m this afternoon.*

Evaluated on the complete Timebank corpus [14], TAC achieves an F-measure of 82.7%. An inspection of a random selection of 200 Timebank annotations revealed a high number of annotation errors (of nearly 10%). Consequently, the evaluation measures give only an approximate value.

4 Event Extraction

Our event extraction algorithm is based on the dependency structure. The dependency structure is the parsing result of a sentence using *Dependency Grammar* (DG), which consists of a bag of dependency relationships. A dependency relationship [9] is an asymmetric binary relationship between one token (i.e. parent node or head) and the other token (i.e. child node or modifier). The dependency structure is a connected tree of all the tokens of the sentence, where each parent node can have several child nodes, but each child node can only have one parent node. And the main verb (i.e. the predicate) of the sentence is the root of the tree.

The use of dependency structure is motivated by the more information it can provide than shallow processing techniques, as well as its robustness and fast speed in comparison with deep parsing. Compared with syntactic structure (i.e. constitute parsing tree), dependency structure captures the semantic relationship between words other than the buildup process of the sentence.

In this paper, we assume that an event can be either represented by a noun (including nominalizations) or by a verb. The main idea of the *EventExtraction* algorithm is to locate the temporal expression in the dependency tree and then traverse the nodes in the tree to find the nearest verb or noun. Since in most cases, the temporal expression is either a modifier of a noun phrase or a part of a verb phrase modifier (usually the latter is realized as a prepositional phrase). The goal of this procedure is to find the corresponding nouns or verbs which the temporal expressions modify.

Algorithm 1 The *EventExtraction* Algorithm

```
function ExtractEvents(DepStr, TempExp): N∪V
    /* DepStr: dependency structure
       TempExp: temporal expression */
    N ← ExtractNounEvent(DepStr, TempExp)
    V ← ExtractVerbEvent(DepStr, TempExp)
end function

function ExtractNounEvent(DepStr, node): N
    Find node in DepStr
    if node.POS == Noun then
        N ← node;
    else
        N ← ExtractNounEvent(DepStr, node.Parent)
    end if
end function

function ExtractVerbEvent(DepStr, node): V
    Find node in DepStr
    if node.POS == Verb then
        V ← node;
    else
        V ← ExtractVerbEvent(DepStr, node.Parent)
    end if
end function
```

Although the algorithm is simple, it works very well in the experiments (cf. Sec. 6). For instance, consider the following **T-H** pair,

(3) <**T**> Released in *1995*, Tyson returned to boxing, winning the World Boxing Council title in *1996*. The same year, however, he lost to Evander Holyfield, and in a *1997* rematch bit Holyfield's ear, for which he was temporarily banned from boxing.

<**H**> In *1996* Mike Tyson bit Holyfield's ear.

The dependency structure of this example is shown partially in Figure 3. After applying our algorithm, the following events could be extracted from **T**.

<**T**>	1995: released (verb)
	1996: winning (nominalization)
	1997: rematch(noun), bit (verb)
<**H**>	1996: bit (verb)

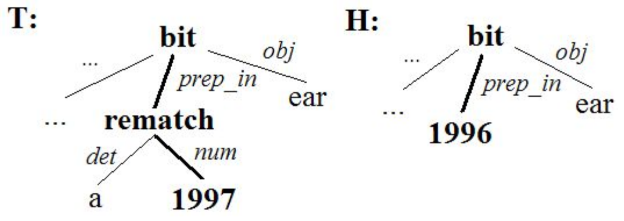

Figure 3. Dependency Structure of Example 3 (shown partially)

5 Recognizing Textual Entailment

Provided by the previous TAC system and event extraction algorithm, we can derive a new feature representation from the input textual pairs. Instead of computing the surface string similarity, we now compare two pairs of temporal expressions and their corresponding events. Such pairs are defined as *EventTime-Pairs* (ETPs), and each of them consists of a noun or a verb denoting the event and the corresponding temporal expression. In order to resolve the relation between two ETPs, we need to separately resolve the relation between events and between temporal expressions, and combine the results afterwards. In the following, we first introduce the lexical resources we have applied, then the entailment rules between two temporal expressions, and finally the complete entailment rule representation.

Lexical Resources In order to acquire the relation between two events (i.e. nouns or verbs), we need lexical resources. WordNet [12] has been widely applied to the RTE task (e.g. [3]). Usually, it is used to discover semantic relations between nouns, e.g. the hypernym/hyponym relation. In our approach, two other features provided by WordNet are considered: 1) the derived form of a noun or a verb; and 2) *entailment* or *entailed-by* relation of a verb. In **H**s, events are usually represented by verbs, except for those cases where *be* is recognized as the main predicate. To improve the coverage of verbs in WordNet, VerbOcean [2] is also used to detect verb relations. In practice, we take *happens-after*, *stronger-than*, and *similar-to* relations together with the equal relation as monotonic relations to the entailment relation. The procedure is as follows,

- verbalize all the nouns, and convert all the nominalizations back to the original verb forms, e.g. *election* to *elect*, *winning* to *win*,
- detect possible relations between verbs, e.g. *win happens-after contest*,

- if at least one above-mentioned relation exists, the entailment between events holds; otherwise, it does not hold.

Entailment rules between temporal expressions Relations between temporal expressions have been discussed a lot by researchers. In particular, TimeML has proposed 13 relations to indicate relations between temporal expressions or between a temporal expression and an event. For our purpose three of them are mapped into the entailment relation. The different granularities and types of temporal expression pairs are also taken into consideration. Consequently, the possible entailment relations between two temporal expressions are shown in Table 1[5].

	P → P	P → D	D → P	D → D
Same	IDENTITY	NO	INCLUDE	INCLUDE
F → C	INCLUDED	NO	INCLUDE	INCLUDE
C → F	NO	IDENTITY	INCLUDE	INCLUDE

Table 1. Entailment Rules Between Temporal Expressions

Rule Representation In Table 2, we define where the relation between a pair of ETPs holds depending on the relations between events and between temporal expressions.

	Event YES	Event NO
Time YES	unknown	NO
Time NO	NO	NO

Table 2. Entailment Rules Between ETPs

Even if both events and temporal expressions hold the entailment relation, other factors can still change the entailment of two ETPs, e.g. the different participants of the events. Since this is out of the scope of this paper, "unknown" is given here as heuristics to the further processing. The other three cases predicate the false entailment relation. Once entailment relations between ETPs in a sentence are found, these relations can be combined so as to determine the entailment relation between sentences, i.e. **T** and **H**. Thus, if the entailment does not hold for all of the ETP pairs, it does not hold for the **T**-**H** pair either; otherwise it is unknown.

To make the process more efficient, we start from **H** to **T**, which is the opposite direction of the entailment relation [18]. The motivations are: **H** is the target we need to examine; **H** is usually simpler than **T**.

Consider Example (3) again, from **H** we can extract an ETP, "<bit, 1996>". In most cases, the event in **H** is represented by a verb, except for sentence like "*The election was in 1992*". To deal with such cases, we manually construct a stop word list containing all the forms of *be* verbs. Together with the ETPs extracted from **T** (shown in Sec. 4), we can compare the following pairs of ETPs,

- <release, 1995>, <bit, 1996> ⟶ NO
- <win[6], 1996>, <bit, 1996> ⟶ NO
- <rematch, 1997>, <bit, 1996> ⟶ NO
- <bit, 1997>, <bit, 1996> ⟶ NO

Therefore, in this **T**-**H** pair, **T** does not entail **H**.

To sum up, the assumption here is that if all the ETPs of **T** do not entail all the ETPs in **H**, the entailment does not hold between **T** and **H**; otherwise, the answer depends on other information. However, in the current system we simplify this problem and consider the latter cases as YES.

6 Evaluation

In order to evaluate our approach, we extract a subset of all data from RTE-2[7] and RTE-3[8]. Table 3 shows the numbers of **T**-**H** pairs containing temporal expressions either in both **T** and **H**, only in **T**, only in **H**, or in neither of them. Table 4 calculates the frequency of time points and durations.

In addition, a data set from TREC 2003 is semi-automatically constructed. The questions and corresponding answers have been used for constructing **H**s and the supporting documents for **T**s. For instance, we combine the question, "*What country made the Statue of Liberty?*" and the answer "*France*" into a statement as **H**, "*France made the Statue of Liberty*". **T** can take the (ir)relevant documents, e.g. "*In 1885, Statue of Liberty arrives in New York City from France*". Finally, we have constructed 313 **T**-**H** pairs (cf. Table 3 and Table 4).

6.1 Experiments

We setup several experiments to evaluate different aspects of our TACTE system. In the first experiment,

[5]P refers to *time points*, D refers to *duration*, F and C refer to fine and coarse granularity respectively. NO means no entailment; otherwise, the entailment holds.

[6]After applying lexical resources to change the nominalization back into the original verb form

[7]http://www.pascal-network.org/Challenges/RTE2

[8]http://www.pascal-network.org/Challenges/RTE3

	RTE-2 dev	RTE-2 test	RTE-3 dev	RTE-3 test	TREC2003	ALL
Both	87 (10.89%)	76 (9.50%)	72 (9.00%)	58 (7.25%)	34 (10.86%)	327 (8.36%)
OnlyT	255	291	275	275	100	1196
OnlyH	15	2	10	8	3	38
Neither	442	431	443	459	176	1951
Total	799	800	800	800	313	3912

Table 3. Statistics of the Data Sets

	RTE-2 dev	RTE-2 test	RTE-3 dev	RTE-3 test	TREC2003	ALL
Time point	191	195	209	155	86	836
per pair	2.20	2.57	2.90	2.67	2.53	2.56
Duration	37	18	15	12	4	86
per pair	0.43	0.24	0.21	0.21	0.12	0.26

Table 4. Statistics of the Temporal Expressions

we compare our system with a *Bag-of-Words* (BoW) system on the data set we extract (Table 5). The BoW approach assigns a similarity score to each **T-H** pair by calculating the ratio between the number of overlapping words in **T** and **H** and the total number of words in **H**. Later a machine learning method SMO [13] in Weka [19] is used to perform a binary classification. This approach is shown to be a very strong baseline for the RTE task on the current data sets [18]. The dependency parser we use is the Stanford Parser [10].

Compared with the BoW baseline system performance on the complete data sets (cf. row 1 in Table 6) with ten-fold cross-validation, the low accuracy shown in row 1 in Table 5 indicates that the **T-H** pairs containing temporal expressions are more difficult. The great improvements (appx. 21% to 49% on different data sets) of the TACTE system shows the advantage of our strategy combining temporal expression anchoring with event extraction.

In order to find out the contribution of the lexical resources, we turn off this part and the third row in Table 5 shows the results. It turns out that the lexical resources do not contribute a lot to the whole system. The largest improvement is on the TREC2003 data set. As an average, this part improves the system with app. 2.5%. The reason is that in these **T-H** pairs with temporal expressions, the respective events in **T** and **H** can be easily distinguished. The limited coverage of our lexical resources is another reason. More work on the lexical semantics is necessary, which corresponds to the results of other approaches, e.g. [4].

We also try to integrate a BoW system into our TACTE system, and there are two ways: either we take the output of our main system as a feature in the machine learning procedure, or leave the BoW system to deal with those **T-H** pairs where not both **T** and **H** contain temporal expressions. The additional feature for the former case would be a ternary value: YES, NO, or UNKNOWN. UNKNOWN is for those cases where not both **T** and **H** contain temporal expressions. Table 6 shows the results. The last two columns give results of training on the development sets and testing on the test sets.

Since the **T-H** pairs with temporal expressions only cover a small proportion (8.36% in Table 3) of the complete data sets, the improvement on the complete data set is less obvious. The results shown in second row in Table 6 is not much different than the baseline, indicating that a systematic feature selection is necessary for the machine learning.

6.2 Error Analysis

In this part we give a detailed error analysis on one of our data sets, i.e. a subset of the RTE-2 development set containing temporal expressions in both **T** and **H**. This subset contains altogether 87 **T-H** pairs, and our TACTE correctly recognizes 67 pairs. Table 7 gives the "error distribution" of the 20 incorrect pairs.

The first kind of errors containing three **T-H** pairs is due to TAC. One error is from SProUT which recognizes "*Today*" in the magazine name "*USA Today*" as a temporal expression. Such an error falsely triggers our anchoring system. Another two errors are implicit temporal expressions introduced by relative clauses and gerunds. In the example "*an incident in 1997, when an enraged Mike Tyson bit Holyfield's ear*", the relative clause introduced by *when* implies that the *bit* event should occur in the same year as *1997*. However, such features cannot be captured and used by our current TAC.

	RTE-2 dev	RTE-2 test	RTE-3 dev	RTE-3 test	TREC2003	Average
BoW (Baseline)	28.74%	46.05%	40.28%	41.38%	26.47%	37.31%
TACTE	**77.01%**	**68.42%**	**61.11%**	**65.52%**	**64.71%**	**68.20%**
TACTE (no LexRes)	74.71%	67.11%	61.11%	63.79%	52.94%	65.75%

Table 5. Experiments Results on Data Containing Temporal Expressions

	RTE-2 dev	RTE-3 dev	TREC2003	RTE-2	RTE-3
BoW (Baseline)	62.58%	67.00%	73.16%	57.88%	61.13%
TACTE + BoW (feature)	62.83%	67.13%	72.52%	58.25%	61.25%
TACTE + BoW (rule)	67.71%	68.88%	76.04%	60.00%	62.88%

Table 6. Experiments Results on the Complete Data Sets

The second kind of errors is due to the RTE system, which contains two subgroups, the parsing part and the event extraction part. We do not discuss the parsing part, since it is out of this paper's scope. All of the three errors coming from the event extraction part are due to the wrong selection of the corresponding events. We also tried to extract more possible events, but it resulted in more ambiguity and the performance decreased. For example, in one **T-H** pair, **T** says "*... after his landslide victory in Sunday's presidential election*", and **H** hypothesizes that person has won the "*Sunday's presidential election*". Although it is correct to relate "*Sunday*" with "*election*", the key events here concerning the entailment relation should be "*victory*" and "*won*".

Lexical resources also bring errors. For instance, there is no relation found between "*was founded*" and "*was opened*". Another example is the lack of relation between "*occur*" and "*fall on*" in "*the Chinese New Year occurred on*" some day and "*the Chinese New Year's Day falls on*" that day.

For the last kind of errors we have not found straightforward solutions yet. Some examples contain complex lexical semantics, e.g. someone "*gave up his throne*" entails he "*abdicated*". Another more difficult example is that "*the blast observed on Dec. 27 came from ...*" entails "*the December burst came from ...*". Not only the lexical relation between "*blast*" and "*burst*" should be known, but also "*observed*" implying that the event followed (i.e. "*came*") also happens in the observation time should be known.

7 Related Work

A number of systems with similar goals as TAC have been developed. The semantic tagging system presented by [15] tries to anchor both time-denoting expressions and event-denoting expressions in German news messages. Event-denoting expressions are more difficult to detect and anchor. The authors admit that only a small set of such expressions can be solved. [6] presented a temporal expression anchorer (TEA), which anchors the temporal expressions in English text and tries to capture their intended meanings. The TEA system was tested on an email dataset with about 150 emails and 279 temporal expressions, and achieves 76.34% for accuracy over the test data set, i.e. the number of correctly anchored expressions over the total number of temporal expressions.

On the other hand, some researchers working on the RTE task also take temporal expressions into consideration. [4] extract and resolve temporal expressions, and use them as features in their approach. However, their system performance is hardly decreased when these features are excluded. This is consistent with our results mentioned in the second row of Table 6. [7] also use temporal expressions in the machine learning procedure. However, there is no separate evaluation showing how much those features contribute to the final results. [17] integrate temporal axioms in their rule-based logic inference system. To some extent these axioms are similar to our entailment rules between temporal expressions. Whereas the pure rule-based system lacks of robustness, if it is not combined with a statistical backup strategy. Compared with their approaches, our TACTE system first concentrates on those cases containing temporal expressions, dealing with the whole RTE problem in a more systematic way.

8 Conclusion and Future Work

In this paper, we have presented our work using a time anchoring system to assist a baseline RTE system on those entailment cases with temporal expressions. After extracting and anchoring the temporal expressions, our TACTE system takes them as starting points in dependency structures and searches for events corresponding to these expressions. With the help of

	Extraction	Anchoring	Parsing	Event Extraction	Lexical Resources	Others
Errors	1	2	5	3	3	6
Percentage	5%	10%	25%	15%	15%	30%

Table 7. Error Distribution

the entailment rules and lexical resources, the entailment relation can be detected. Several experiments on various data sets are conducted, and TACTE shows a significant improvement on the baseline system.

For the future development, on the one hand we consider to extend the *OWLTime* ontology to include repetitive temporal expressions, such as *every Wednesday*, so as to improve the coverage of the anchoring system. On the other hand, we consider to integrate the TACTE into a larger framework so that more entailment cases can be handled.

Acknowledgments

This work was supported partially by the European Commission under contract number FP6-027685 (MESH) and the BMBF project HyLaP (FKZ: 01 IW F02).

References

[1] J. F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26:832–843, 1983.

[2] T. Chklovski and P. Pantel. Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP-04)*, volume 34, pages 343–360, Barcelona, Spain, 2004.

[3] I. Dagan, O. Glickman, and B. Magnini. The PASCAL Recognising Textual Entailment Challenge. In *Machine Learning Challenges*, volume 3944 of *Lecture Notes in Computer Science*, pages 177–190. Springer, 2006.

[4] M.-C. de Marneffe, B. MacCartney, T. Grenager, D. Cer, A. Rafferty, and C. D. Manning. Learning to distinguish valid textual entailments. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy, 2006.

[5] W. Drozdzynski, H.-U. Krieger, J. Piskorski, U. Schäfer, and F. Xu. Shallow processing with unification and typed feature structures — foundations and applications. *Künstliche Intelligenz*, 1:17–23, 2004.

[6] B. Han, D. Gates, and L. Levin. From language to time: A temporal expression anchorer. *Proceedings of the Thirteenth International Symposium on Temporal Representation and Reasoning (TIME'06)*, pages 196 – 203, 2006.

[7] A. Hickl, J. Williams, J. Bensley, K. Roberts, B. Rink, and Y. Shi. Recognizing Textual Entailment with LCC's Groundhog System. In *Proceedings of the Second PASCAL Recognizing Textual Entailment Challenge*, Venice, Italy, 2006.

[8] J. R. Hobbs and F. Pan. Time Ontology in OWL. W3C Working Draft 27 September 2006. http://www.w3.org/TR/2006/WD-owl-time-20060927/, 2006.

[9] R. Hudson. *Word Grammar*. Basil Blackwell Publishers Limited, Oxford, England, 1984.

[10] D. Klein and C. D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, pages 423–430, 2003.

[11] D. Lin and P. Pantel. Discovery of inference rules for question answering. *Natural Language Engineering*, 34:343–360, 2001.

[12] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. Five papers on WordNet. Technical report, Cognitive Science Laboratory, Princeton University, 1993.

[13] J. Platt. Machines using sequential minimal optimization. *Advances in Kernel Methods - Support Vector Learning*, 1998.

[14] J. Pustejovsky, J. Castano, R. Ingria, R. Saur, R. Gaizauskas, A. Setzer, and G. Katz. TimeML: Robust specification of event and temporal expressions in text. In *Fifth International Workshop on Computational Semantics*, 2003.

[15] F. Schilder and C. Habel. From temporal expressions to temporal information: Semantic tagging of news messages. *Proceedings of ACL'01 workshop on temporal and spatial information processing*, pages 65–72, 2001.

[16] Y. Shinyama and S. Sekine. Paraphrase acquisition for information extraction, 2003.

[17] M. Tatu, B. Iles, J. Slavick, A. Novischi, and D. Moldovan. COGEX at the second recognizing textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy, 2006.

[18] R. Wang and G. Neumann. Recognizing textual entailment using a subsequence kernel method. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (AAAI-07)*, Vancouver, Canada, July 22-26 2007.

[19] I. H. Witten and E. W. Frank. Practical machine learning tools and techniques with java implementations. *Proceedings of the ICONIP/ANZIIS/ANNES*, 1999.