The DANTE Temporal Expression Tagger

Paweł Mazur[1,2] and Robert Dale[2]

[1] Institute of Applied Informatics, Wrocław University of Technology
Wyb. Wyspiańskiego 27, 50-370 Wrocław, Poland
Pawel.Mazur@pwr.wroc.pl
[2] Centre for Language Technology, Macquarie University,
NSW 2109, Sydney, Australia
{mpawel,rdale}@ics.mq.edu.au

Abstract. In this paper we present the DANTE system, a tagger for temporal expressions in English documents. DANTE performs both recognition and normalization of these expressions in accordance with the TIMEX2 annotation standard. The system is built on modular principles, with a clear separation between the recognition and normalisation components. The interface between these components is based on our novel approach to representing the local semantics of temporal expressions. DANTE has been developed in two phases: first on the basis of the TIMEX2 guidelines only, and then using the ACE 2005 development data. The system has been evaluated on the ACE 2005 and ACE 2007 data. Although this is still work in progress, we already achieve highly satisfactory results, both for the recognition of temporal expressions and their interpretation.

Keywords: Temporal expression tagging, semantics, underspecification.

1 Introduction

The task of temporal expression recognition and normalisation involves identifying, within texts, expressions that refer to points or periods of time, and re-expressing these temporal references in a standard format which (a) precisely describes the semantics of the expressions, (b) disambiguates dates and times from different time zones, and (c) makes it easier to determine the sequencing of events described in these texts.

The time expression normalisation task is an interesting and challenging one because, while some temporal references appear in well-defined formats, others are expressed using a wide range of natural language constructions, and are often ambiguous, requiring analysis of the surrounding text in order to arrive at an interpretation. Of course, there are cases where information external to a document—perhaps contained in another document, or best considered part of world knowledge—is required in order to interpret a temporal expression; such cases are not considered here.

Z. Vetulani and H. Uszkoreit (Eds.): LTC 2007, LNAI 5603, pp. 245–257, 2009.

There have always been sections of the linguistics, philosophy and natural language processing communities that have been interested in temporal referring expressions. However, interest in the recognition and interpretation of these expressions has grown significantly as a result of the DARPA-sponsored competitions in named entity recognition from the mid-1990s onwards. In contrast to earlier work in the area, these competitions and related exercises introduced a rigorous evaluation paradigm, whereby success or failure was measured in terms of the ability of software systems to replicate human 'gold standard' annotations of the scope and interpretation of temporal referring expressions.

Undoubtably, the key events and exercises that have played a role in this growth have been the Message Understanding Conferences (MUCs) in 1996 and 1998, and the workshops associated with the Automatic Content Extraction (ACE) program[1] in 2004, 2005 and 2007. While both MUC evaluations covered only recognition of two types of temporal expressions (dates and times), there has been a significant increase in the level of task difficulty in the ACE competitions. The fundamental move forward here was the addition of a normalisation task to the recognition task: annotations were provided for the interpretation of dates and times by using TIMEX2, a slightly modified version of ISO 8601, as the standard for the representation of normalized dates and times. The introduction of TIMEX2 also influenced the recognition task, as the range of temporal expressions to be recognised was broadened significantly as compared to the MUC-6 and MUC-7 task definitions.

Subsequently, the TIMEX2 standard has evolved through a number of versions, partially due to the wide interest it has received in the community, and the existence of the ACE program and similar competitions. This has also resulted in quite a large number of temporal expression taggers being constructed by the participants in these competitions. Details of the current, and most likely final, version of the TIMEX2 standard are provided in [1].

In this paper we present the DANTE (Detection And Normalisation of Temporal Expressions) system, which, as its name suggests, performs both recognition and normalisation of temporal expressions. Currently, the system works only for English texts; however, its extension to other languages is facilitated by its modular architecture, where some components are language independent. In January 2007, DANTE participated in the ACE Time Expression Recognition and Normalization (TERN) task.

The rest of this paper is organized as follows. First, in Section 2 we briefly introduce the TIMEX2 annotation schema. In Section 3 we describe related work, briefly presenting other existing temporal expression taggers. Then, in Section 4, we outline our model for the representation of temporal expressions. Section 5 presents the processing model we have adopted, and, in Section 6, DANTE's system architecture and development process is discussed. Section 7 provides information on DANTE's performance both in terms of recognition and normalisation results, and in terms of resource consumption and execution time. Conclusions and future work are discussed in Section 8.

---

[1] See `http://www.nist.gov/speech/tests/ace`

Table 1. Attributes in TIMEX2

Attribute	Description
`VAL`	Contains a normalized value in ISO-like format of the date or time of the annotated expression.
`MOD`	Captures temporal modifiers, using values such as `BEFORE`, `AFTER`, `LESS_THAN`, `MORE_THAN`, `EQUAL_OR_LESS`, `START`, `MID`, `END` or `APPROX`.
`ANCHOR_VAL`	Contains a normalized value in ISO-like format of an anchoring date or time.
`ANCHOR_DIR`	Captures the relative direction or orientation between `VAL` and `ANCHOR_VAL` attributes, as in `WITHIN`, `STARTING`, `ENDING`, `AS_OF`, `BEFORE` or `AFTER`. It is used to express information about *when* a duration is placed.
`SET`	Identifies expressions denoting sets of times; either takes the value `YES` or is empty.
`COMMENT`	Contains any comment that the annotator wants to add to the annotation; ignored from the point of view of automatic processing of the text.

2 The TIMEX2 Annotation Schema

The TIMEX2 schema provides an inline SGML tag, `TIMEX2`, for annotating temporal expressions. Annotations can be nested, as shown in Example (1).

(1) I'm leaving on vacation `<TIMEX2 VAL="1999-08-03">`two weeks from `<TIMEX2 VAL="1999-07-20">`next Tuesday`</TIMEX2></TIMEX2>`.

There are six attributes defined in the schema for the tag; values of attributes express the semantics of an expression. A description of the attributes is given in Table 1. In this context, the recognition task is concerned with finding the boundaries of temporal expressions in texts, and the normalisation task involves determining the values of TIMEX2 attributes associated with this temporal expression.

3 Related Work

The earliest approaches, typical of work undertaken for MUC-6, were based on the construction of hand-crafted rules using a grammatical formalism that would match both fixed-format dates and times, and a range of expressions in natural language within the scope defined in the guidelines. For MUC-7, there were both solutions based on transducers, such as those described in [2] and [3], and also other techniques, such as Hidden Markov Models as used in IdentiFinder [4][2]. In both MUC competitions, the results achieved for TIMEX recognition by the best systems were high:

- at MUC-6, Recall of 93% and Precision of 96% were reported; and
- at MUC-7, Recall of 89% and Precision of 99% for dates, and Recall of 81% and Precision of 97% for times were reported [3].

[2] See also [5] for an extended description.

Table 2. The F-measure results for ATEL, Chronos and GUTime on ACE TERN 2004 data. ATEL and Chronos were evaluated on the ACE TERN 2004 evaluation data, while GUTime's performance was measured on the ACE TERN 2004 training corpus. All results cited here are from the original papers by the authors of the systems.

	Detection	Extent Recognition	`VAL` Attribute
ATEL	93.5	87.8	–
Chronos	92.6	83.9	87.2
GUTime	85	78	82

TempEx (see [6]) was the first TIMEX2 tagger developed. It is a relatively simple Perl tool that implements a few heuristics based on part-of-speech tags using finite state automata. It also performs limited normalisation of the expressions. The most recent version, from December 2001, implements the 2001 version of the TIMEX2 standard. There are certain classes of phrases that are not recognized by this tool: for example, *the last Monday of January*, *the end of 1999*, and *late yesterday morning*. This tool was provided to all participants of ACE TERN 2004 for use as an external source of text features; as such, it provides a reasonable baseline for performance on new data.

GUTime [7] was developed as an extension of TempEx for the purpose of constructing an automatic temporal annotation tool for TimeML (see [8]). TimeML is a sophisticated schema for the annotation of events; its complexity means that automatic tagging of events is best achieved via a cascade of modules that successively add more and more TimeML annotations to the document being processed. In this context, GUTime is the module responsible for the detection of temporal expressions and the introduction of the `TIMEX3` tag into the annotations[3]. GUTime's coverage of temporal expressions is greater than that of TempEx. In addition, it also handles TIMEX3's functional approach to expressing values: that is, for relative expressions it first identifies what function is realised by an expression (for example, for *tomorrow* it would be PLUS ONE DAY), so that the actual value of that function (for example, *25th January 1996*) can be calculated at a later stage.

Chronos [9] is a more complex system designed to perform both recognition and normalisation of temporal expressions. Text processing in Chronos involves tokenization, statistical part-of-speech tagging and recognition of multiword elements based on a list of 5000 entries retrieved from WordNet. Then, the text is processed by a set of approximately 1000 basic rules that recognize temporal constructions and gather information about them that is expected to be useful in the process of normalization. This is followed by the application of composition rules, which resolve ambiguities when multiple tag placements are possible.

[3] TIMEX3 is part of the TimeML schema. It is very similar to TIMEX2, but provides a different approach to the annotation of sets and durations in time expressions and introduces the notion of temporal functions as means of providing the value of an expression.

The increasing availability of corpora annotated with temporal expressions[4] makes it possible to apply supervised machine learning techniques to the time expression recognition problem. Examples of such systems are ATEL [10], TimexTag[5] [11] and Alias-i's LingPipe[6]. ATEL and TimexTag are based on Support Vector Machine (SVM) classifiers, and LingPipe is constructed using a Hidden Markov Model (HMM). While ATEL only carries out the recognition of temporal expressions, TimexTag also performs interpretation, via 89 interpretation rules used by the interpreting SVM-based modules.

Table 2 presents the performance of a number of systems on the ACE 2004 TERN data. An annotation produced by a system counts as a correctly *detected* temporal expression if there is a gold standard annotation which which it overlaps by at least one character. A correctly *recognized* expression is one for which the extent produced by a system is exactly the same as in the gold standard. Evaluation of the `VAL` attribute involves checking that the correct value of this TIMEX2 attribute has been generated.

4 Representing Temporal Expressions

As is conventional in this area of research, we view the temporal world as consisting of two basic types of entities, these being **points in time** and **durations**; each of these has an internal hierarchical structure. We find attribute–value matrices to be a good means of clearly representing the semantics of temporal expressions. Figure 1 shows the representation of a reference to a specific point in time; in the ISO date and time format used in the TIMEX2 standard, this would be written as follows:

(2) `2006-05-13T15:00:00Z`

Each atomic feature in the attribute–value structure thus corresponds to a specific position in the ISO format date–time string.

Of course, very few of the temporal expressions normally found in text are so fully specified. The attribute–value matrix representation we use makes it very easy to represent the content of underspecified temporal expressions. For example, the content of the temporal expression *13th May* in a sentence like *We will meet on 13th May* can be expressed as follows:

(3)
$$\begin{bmatrix} \text{point} & \\ \text{timeanddate} & \begin{bmatrix} \text{date} & \begin{bmatrix} \text{day} & \begin{bmatrix} \text{daynum} & 13 \end{bmatrix} \\ \text{month} & 05 \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

[4] See `http://timexportal.wikidot.org` for an extensive list of existing corpora with annotations of temporal expressions.

[5] See `http://ilps.science.uva.nl/resources/timextag`

[6] See `http://www.alias-i.com/lingpipe`

$$\begin{bmatrix} \text{point} & \\ \text{timeanddate} & \begin{bmatrix} \text{time} & \begin{bmatrix} \text{hour} & 15 \\ \text{mins} & 00 \\ \text{secs} & 00 \end{bmatrix} \\ \text{date} & \begin{bmatrix} \text{day} & \begin{bmatrix} \text{dayname} & \text{D4} \\ \text{daynum} & 13 \end{bmatrix} \\ \text{month} & 5 \\ \text{year} & 2006 \end{bmatrix} \end{bmatrix} \\ \text{zone} & \text{Z} \end{bmatrix}$$

Fig. 1. The semantics of the expression *3pm Thursday 13th May 2006 GMT*

The values of the atomic elements in such an expression come from the lexicon, and the value of a composite structure is produced by unifying the values of its constituents. Here, the lexical entry for *13th* delivers the structure in Example (4a), and the lexical entry for *May* delivers the structure in Example (4b):

(4) a. $$\begin{bmatrix} \text{point} & \\ \text{timeanddate} & \begin{bmatrix} \text{date} & \begin{bmatrix} \text{day} & \begin{bmatrix} \text{daynum} & 13 \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

b. $$\begin{bmatrix} \text{point} & \\ \text{timeanddate} & \begin{bmatrix} \text{date} & \begin{bmatrix} \text{month} & 05 \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

When unified, these structures produce the representation shown in Example (3). Multiword sequences, such as idioms, that are best considered atomic and not compositional in their semantics can be assigned semantic representations directly in the lexicon.

5 The Processing Model

We take the view that an important step towards a truly broad-coverage yet semantically-well-founded approach is to recognize that there is a principled distinction to be made between the interpretation of the semantics of a temporal expression devoid of its context of use, and the fuller interpretation of that expression when the context is taken into account. The first of these, which we refer to here as the **local semantics** of a temporal expression, should be derivable in a compositional manner from the components of the expression itself; determining the value of the second, which we refer to as the **global semantics** of the expression, may require arbitrary inference and reasoning. Such a distinction is implicit in other accounts: Schilder's [12] use of lambda expressions allows representation of partially specified temporal entities, and the temporary variables that Negri and Marseglia [9] construct during the interpretation of a given temporal expression capture something of the same notion as our local semantics.

The above assumptions are reflected in our design, which comprises separate and independent modules for the recognition and normalisation subtasks, with the first being responsible for the computation of the local semantics, and the second being responsible for determining the global semantics.

We assume a **granularity ordering** over what we might think of as the **defining attributes** in a temporal representation:

(5) year > month > daynum > hour > minute > second

These are, of course, precisely the elements that are represented explicitly in an ISO date–time expression.

Interpretation of a partially specified temporal expression then requires ensuring that there is a value for every defining attribute that is of greater granularity than the smallest granularity present in the partially specified representation. We refer to this as the **Granularity Rule** in interpretation.

In the case of Example (3) above, the Granularity Rule tells us that in order to compute the full semantic value of the expression we have to determine a value for YEAR, but not for HOUR, MINS or SECS. This interpretation process may require various forms of reasoning and inference, and is qualitatively different from the computation of the local semantics.

6 System Architecture and Development

In our system, the stages of text processing are organized as a pipeline of processing resources, run using the architectural constructs provided in GATE [13]. The elements in our pipeline are a tokenizer, gazetteers, a sentence splitter, a POS tagger, named entity recognition, temporal expression recognition, and temporal expression interpretation[7].

6.1 Temporal Expression Recognition

The temporal expression recognizer is implemented using a JAPE grammar. The grammar consists of five phases which are run over a document in sequence. Each phase contains rules which match annotations introduced by earlier processing components (for example, the tokenizer or POS tagger) and JAPE grammar phases. There is also one initial additional phase which consists only of the expansion of macros used in the grammar rules. Altogether there are 81 macros and 252 rules; macro expansions are textually copied into the bodies of rules, and then the rules are compiled into Java code. Some of the rules are *negative rules*; they

[7] We refer to this here as an "interpreter" since what is really happening in the "normalisation" process is in fact the interpretation of a temporal expression in the context of the rest of the document.

do not generate any annotations and are used only to prevent the positive rules to match expressions which are not time-referring, but which are similar in their appearance.

JAPE rules are traditional pattern–action rules, where the left-hand side contains the pattern to be matched, and the right-hand side specifies the action to be taken when the pattern is matched. The pattern on the left-hand side is written using JAPE syntax, but the right-hand side can be implemented either in JAPE or directly in Java code. Our recognition rules use 33 gazetteers with a total of 1354 entries: these are strings used in the expression of dates and times, such as numbers written in words; the names of days, months and time zones; and the most common fractions (as used in, for example, *one third* or *1/2*).

The development of our temporal expression recognition module was carried out in a number of steps. The first of these took two and a half person months; the module was developed on the basis of the TIMEX2 guidelines and the examples contained therein. Then we tested DANTE on the ACE 2005 development data and identified frequently-occurring cases which were problematic for the system. Addressing these problems constituted a second stage of system development. Subsequent development has focussed on fixing bugs and other similar errors discovered as we have applied the system to an ever-wider range of document data sources; the system's grammatical coverage has remained relatively stable.

6.2 Local Semantics Encoding

Attribute-value matrices are relatively unwieldy in implementation as compared to the simple string structures used as values in the TIMEX standard; in particular, they are not usable as input to existing system evaluation tools. So, to enable easy evaluation of a system's ability to construct the intermediate semantic representations that correspond to our notion of local semantics, we would like to use a representation that is immediately usable by existing evaluation tools. To achieve this goal, we define a number of extensions to the standard TIMEX2 string representation for values of the VAL attribute; these extensions allow us to capture the range of distinctions we need to capture the different kinds of underspecification that are present only in our local semantics representation. Tables 6.3 and 6.3 present examples of string encodings of the local semantics of underspecified and relative temporal expressions. These type of strings are found in the output of the system's recognition module. See [14] and [15] for more details and examples of this representation.

6.3 Temporal Expression Interpretation

The interpreter module is a process that steps through a document sentence by sentence. Each temporal expression identified in the recognition stage is passed through the interpretation module, which transforms the local semantic representation into a document-internal semantic representation. The interpreter is fully

Table 3. Underspecified dates and times in ISO-like format

#	String	Representation	#	String	Representation
1	9 pm	xxxx-xx-xxT21	6	the nineteenth	xxxx-xx-19
2	11:59 pm	xxxx-xx-xxT23:59	7	January 3	xxxx-01-03
3	eleven in the morning	xxxx-xx-xxT11:00	8	November	xxxx-11
4	ten minutes to 3	xxxx-xx-xxt02:50	9	summer	xxxx-SU
5	15 minutes after the hour	xxxx-xx-xxtxx:15	10	the ’60s	xx6

Table 4. Relative dates and times in ISO-like format

#	String	Representation	#	String	Representation
1	today	+0000-00-00	6	sixty seconds later	+0000-00-00T+00:00:60
2	tomorrow	+0000-00-01	7	five minutes ago	+0000-00-00T−00:05
3	yesterday	−0000-00-01	8	in six hours time	+0000-00-00T+06:00
4	last month	−0000-01	9	at 6 a.m. today	+0000-00-00T06:00
5	three years ago	−0003	10	last night	−0000-00-01TNI

implemented in Java and includes a library of functions for various calculations on dates and times. This module took approximately one and a half person months to develop.

In our current model, we assume that a document has a simple linear structure, and that any hierarchical structure in the document has no bearing on the interpretation of temporal expressions; for present purposes we also make the simplifying assumption that the **temporal focus** used to compute document-level values for temporal expressions does not advance during the processing of the document. Both assumptions may not always hold true, but are likely to work for the majority of cases we are dealing with.

Depending on the type of the temporal expression being interpreted (fully specified point in time, underspecified point in time, relative expression, duration, frequency and so on), different actions are taken. The two basic operations used in the interpretation are unification with some reference date and the addition or subtraction of a specified number of units to or from a reference date. The type of the temporal expression is also important for determining which TIMEX2 attributes other than VAL should be generated.

6.4 Cycle-Based Calendar Temporal Expression Interpretation

A distinctive class of relative temporal expressions are bare expressions based on calendar cycles—i.e., weekday names and month names—are used, as in the following example:

(6) Jones met with Defense Minister Paulo Portas on *Tuesday* and will meet Foreign Minister Antonio Martins da Cruz before leaving Portugal *Wednesday*.

Here, the proper interpretation of the references to *Tuesday* and *Wednesday* requires at the least a correct syntactic analysis of the sentence, in order to locate the **controlling verb** for each weekday name. The tense of this verb can then be used to determine the direction—either in the past or in the future—in which we need to look to establish the fully specified date referred to. In the case of Example (6), this means determining that *Tuesday* is in the scope of the verb *met*, and that *Wednesday* is in the scope of the verb group *will meet*. It turns out, however, that there are cases where even the controlling verb does not provide sufficient information to determine the 'direction of offset'. But even in those cases where the tense of the verb *does* provide the relevant information, there are two problems. First, especially when the sentences considered are complex, there is a non-negligible likelihood that the analysis returned by a parser may not be correct, and this is especially the case when the sentences in question contain structures such as prepositional phrases: the attachment of these is notoriously a source of ambiguity, and they just happen to often be the hosts to temporal expressions. Second, even if a parser provides the correct analysis, parsing technology is still computationally expensive to use when processing very large bodies of text; if we are interested in time-stamping events described in significant volumes of data, we would prefer to have a faster, more heuristic-based approach.

In [16] we explored the development of a fast and high accuracy algorithm for the interpretation of weekday names, in particular with regard to determining the direction of offset to be used in the temporal interpretation of these expressions: in essence, how can we determine whether the day referred to is in the past or in the future? In that work, we described an approach, based on the algorithm presented by Baldwin in [17], that achieves 95.91% accuracy on a data set created from the ACE 2005 Training corpus, outperforming a range of other approaches we considered.

7 System Performance

7.1 Evaluation

The most significant evaluations of DANTE to date result from our participation in the ACE 2007 TERN task, and our subsequent re-evaluation of the system on the same data after further development on the ACE 2005 development data set.

In the ACE evaluations, a correctly recognized time expression is one which has a strictly accurate extent, and correct values for all the TIMEX2 attributes. An annotation generated by a system is classified as matched with an annotation from the gold standard if there is minimum 30% text span overlap between them

The ACE 2007 evaluation data included 2028 time expressions to be recognized and interpreted, spread across 254 documents from six different domains (see Table 5). As one might expect, documents were not equally distributed across the domains, both in terms of the number of documents and the total size. Across all domains, we currently achieve 74.19%, 77.66% and 75.89 for precision, recall and F-measure, respectively, for correct identification of extents of temporal expressions in text. Detailed results are shown in Table 5. The column titled

Table 5. The results of the DANTE system on the ACE 2007 evaluation data set

Domain	TIMEX2 in Gold Standard	Spurious	Missing	Error	ACE Value
Broadcast Conversations	142	33	29	48	43.8
Broadcast News	322	99	38	66	55.9
Newswire	894	125	110	221	63.0
Telephone Conversations	70	20	11	25	51.3
Usenet Newsgroups	167	20	22	42	65.5
Weblogs	433	66	58	137	57.9
Total	2028	363	268	537	59.1

'Spurious' indicates the numbers of false positive cases, i.e. when the system generated an annotation that is not present in the gold standard. The column titled 'Missing' presents the numbers of false negative cases, which are those cases when an annotation in the gold standard cannot be matched with any annotation in the system output. A matched annotation counts as an error if its textual extension or the value of any of the attributes is wrong. The ACE value score, which is shown in the last column, is an evaluation metric used in the ACE evaluations; it is defined to be 'the sum of the values of all of the system's output TIMEX2 tokens, normalized by the sum of the values of all of the reference TIMEX2 tokens. The value score attributed to each system token is based on its attributes and on how well it matches its corresponding reference token.' The attributes and the extent are weighted,[8] to allow for a difference in importance of the various attributes. The overall ACE TERN value for DANTE is 59.1 (out of 100), which indicates that DANTE's performance is in the range of state-of-the-art systems.

7.2 Resource Utilisation

The execution time for our text processing modules is presented in Table 6 as measured on a laptop with a 2GHz Intel Core Duo processor; however, only one core of the processor was used for processing documents. In characterising the processing cost, we do not take into account initialization of the system, the exporting of results into XML files, and the postprocessing required to meet the ACE formatting requirements, including the conversion of results from our inline XML annotation into the APF XML format.

Memory consumption during system execution is to some extent dependent on the size of the processed document, but on the ACE 2007 evaluation the variation was not great (from 116MB to 126MB). The system also required approximately 15MB of disk space to store the input corpus. As noted above, documents were not

[8] In the ACE 2007 TERN evaluations the weights were as follows: 1.0 for type VAL, 0.5 for ANCHOR_VAL, 0.25 for ANCHOR_DIR, 0.1 for MOD, 0.1 for SET, and 0.1 for extent (at least 0.3 overlap between matched elements, otherwise elements are not considered matched at all). The cost of a spurious TIMEX2 mention was 0.75 of the sum of the weights for the extent and for all generated attributes of the annotation.

Table 6. Execution times on the ACE 2007 eval data set

Domain	# docs	Time [s]	Av. time per doc [s]	Size [B]	Av. time/10kB [s]
Broadcast Conversations	9	8.698	0.966	48,722	1.828
Broadcast News	74	12.386	0.167	75,731	1.675
Newswire	106	36.306	0.343	209,973	1.748
Telephone Conversations	6	9.306	1.551	54,522	1.771
Usenet Newsgroups	13	9.000	0.692	48,377	1.905
Weblogs	46	23.000	0.500	137,549	1.712
Total	254	98.696	0.389	574,874	1.758

equally distributed across the domains, so we ran the system for each document source type separately in order to identify variations in performance across the different domains. On average, DANTE processed about 5.7KB of input data per second[9].

8 Conclusions and Future Work

In this paper, we have presented our approach to recognition and interpretation of temporal referring expressions in English natural language texts, and the implementation of this approach in the DANTE system. The system has been evaluated on the ACE 2007 evaluation corpus, which is a data set widely accepted by the community as a gold standard for the TERN task. The achieved results are of high enough quality use DANTE in many applications that require the interpretation of temporal expressions in text processing, such as information extraction and question answering.

Our evaluations have brought to light several areas where DANTE can be improved. Our error analysis indicates that the following steps should have the highest priority:

- First, we need to further develop the recognition grammar to cover some rarer cases. This requires both the addition of vocabulary to our existing rules, and also the development of new rules covering previously unseen structures. As the system is rule-based, this also requires careful testing to ensure that the addition or modification of rules does not introduce any incompatibilities or inconsistencies in the grammar.
- Second, we need to improve our mechanism for focus tracking in documents in order to more accurately resolve ambiguities, particularly in extended texts. Although using the document creation date as the temporal focus often works fairly well in simple news texts, it is not reliable enough for temporal expressions tagging across a broader range of text types.

Acknowledgements. We acknowledge the support of the Australian Defence Science and Technology Organisation in carrying out the work described here.

[9] Input data size here is the total size of the input documents including XML tags, which are ignored by the pipeline.

References

1. Ferro, L., Gerber, L., Mani, I., Sundheim, B., Wilson, G.: TIDES 2005 Standard for the Annotation of Temporal Expressions. Technical report, MITRE (2005)
2. Mikheev, A., Grover, C., Moens, M.: Description of the LTG System Used for MUC-7. In: 7th Message Understanding Conference, Fairfax, Virginia (1998)
3. Krupka, G., Hausman, K.: IsoQuest Inc.: Description of the NetOwl(TM) Extractor System as Used for MUC-7. In: 7th Message Understanding Conference, Fairfax, Virginia (1998)
4. Miller, S., Crystal, M., Fox, H., Ramshaw, L., Schwartz, R., Stone, R., Weischedel, R.: The Annotation Group: BBN: Description of the SIFT System as Used for MUC-7. In: 7th Message Understanding Conference, Fairfax, Virginia (1998)
5. Bikel, D., Schwartz, R., Weischedel, R.: An Algorithm that Learns What's in a Name. Machine Learning 34, 211–231 (1999)
6. Mani, I., Wilson, G.: Robust Temporal Processing of News. In: 38th Annual Meeting of the ACL, pp. 69–76. Assoc. for Comput. Linguistics, Morristown (2000)
7. Verhagen, M., Mani, I., Sauri, R., Littman, J., Knippen, R., Jang, S.B., Rumshisky, A., Phillips, J., Pustejovsky, J.: Automating Temporal Annotation with TARSQI. In: ACL Interactive Poster and Demonstration Sessions, pp. 81–84. Association for Computational Linguistics, Ann Arbor (2005)
8. Pustejovsky, J., Ingria, B., Sauri, R., Castano, J., Littman, J., Gaizauskas, R., Setzer, A., Katz, G., Mani, I.: The Specication Language TimeML. In: Mani, I., Pustejovsky, J., Gaizauskas, R. (eds.) The Language of Time: A Reader. Oxford University Press, Oxford (2004)
9. Negri, M., Marseglia, L.: Recognition and Normalization of Time Expressions: ITC-irst at TERN 2004. Tech. Report WP3.7, Information Society Technologies (2005) Information Soc. Technologies (2005)
10. Hacioglu, K., Chen, Y., Douglas, B.: Automatic time expression labeling for english and chinese text. In: Gelbukh, A. (ed.) CICLing 2005. LNCS, vol. 3406, pp. 548–559. Springer, Heidelberg (2005)
11. Ahn, D., van Rantwijk, J., de Rijke, M.: A cascaded machine learning approach to interpreting temporal expressions. In: Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2007), Rochester, NY, USA (2007)
12. Schilder, F.: Extracting Meaning from Temporal Nouns and Temporal Prepositions. ACM Transactions on Asian Lang. Information Processing 3, 33–50 (2004)
13. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A framework and graphical development environment for robust NLP tools and applications. In: 40th Annual Meeting of the ACL, pp. 54–62. Association for Computational Linguistics, Philadelphia (2002)
14. Mazur, P., Dale, R.: An Intermediate Representation for the Interpretation of Temporal Expressions. In: COLING/ACL 2006 Interactive Presentation Sessions, pp. 33–36. Association for Computational Linguistics, Sydney (2006)
15. Dale, R., Mazur, P.: Local Semantics in the Interpretation of Temporal Expressions. In: Workshop on Annotating and Reasoning about Time and Events (ARTE), pp. 9–16. Association for Computational Linguistics, Sydney (2006)
16. Mazur, P., Dale, R.: What's the date? High accuracy interpretation of weekday. In: 22nd International Conference on Computational Linguistics (Coling 2008), Manchester, UK, pp. 553–560 (2008)
17. Baldwin, J.: Learning Temporal Annotation of French News. Masterthesis, Department of Linguistics, Georgetown University (2002)