The University of Amsterdam at ACE 2007: The English TERN task

David Ahn Joris van Rantwijk Maarten de Rijke

1 Introduction

For ACE 2007, the ILPS group from the University of Amsterdam participated in only one task: English TERN. We submitted two runs: the second run differed from the first only in fixing incorrect character offsets due to problems with the empty <UNTRANSLATEDTEXT/> element and SGML character entities.

2 Architecture

The architecture of our timex annotation system is depicted in Fig. 1. Our system begins with parsed documents as input. Our recognition module is a machine learned classifier (A) that classifies syntactic constitutents as timexes or non-timexes. This module is described in §3.

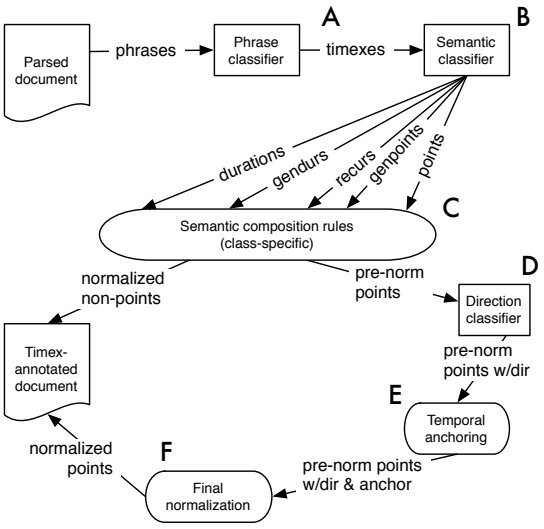

Figure 1: Timex annotation architecture (letters for ease of reference).

Phrases that have been classified as timexes are then sent to the semantic class classifier (B). Semantic class disambiguation is the first point at which context dependence enters into timex interpretation. While some timexes are unambiguous with respect to whether they refer to a point, a duration, or a set, many timexes are semantically ambiguous and can only be disambiguated in context. The machine learned classifier for this task is described in §4.

Based on the semantic class assigned by the semantic class classifier, the semantic composition component (C) of the system generates (underspecified) semantic representations using class-specific, context-independent rules. The rules in our system are simple pattern-matching rules that map lexical items or sequences of lexical items within a timex to semantic representations. We briefly describe the semantic composition component in §5.

For most classes of timexes, the semantic composition component generates a semantic representation that can be directly translated into a normalized value. Timexes that refer to specific points are the only exception. While some point timexes are fully qualified, and thus also directly normalizable, many need to be anchored to another time in context in order to be fully normalized. Thus, context dependence again enters the timex interpretation process, and now in two ways. One is obvious: these referential timexes, which need a temporal anchor, have to find it in context. This task requires a reference resolution process (E), which is described in §6.1.

The second ambiguity regards the relation between a referential timex and its anchor. Referential timexes, like anaphoric definites, relate to their anchors through a bridging relation, which is determined primarily by the content of the timex—e.g.,

two years later refers to a point two years after its anchor. For some referential timexes, though, the direction of the relation (before or after the anchor) is not specified. Resolution of this ambiguity is handed over to the machine learned direction classifier (D), which is described in §6.2.

For referential timexes, final normalization (F) is a straightforward combination of semantic representation, temporal anchor, and direction class.

Not pictured in Fig. 1 is a module that recognizes and normalizes timexes in document metadata using a set of simple regular expressions (REs; 14 in total). This module also determines the document timestamp for referential timexes by using a few heuristics to choose from among multiple timestamps or a date from the document text, if necessary.

While our architecture is novel, we are not the first to modularize timex annotation systems. Even thoroughly rule-based systems, such as (Negri and Marseglia, 2004) and (Saquete et al., 2002), separate temporal anchor tracking from the rest of the normalization process. The system of (Mani and Wilson, 2000) goes further in using separate sets of hand-crafted rules for recognition and normalization and in separating out several disambiguation tasks. (Ahn et al., 2005b) similarly decouple recognition from normalization—even using machine learning for recognition—and also handle several disambiguation tasks separately. In none of these systems, though, are context-independent and context-dependent processing thoroughly separated, as here, and in all these systems, it is the rules that drive the processing—in both Mani et al. and Ahn et al.'s systems, sets of rules are used to determine which timexes need to be disambiguated.

3 Component A: Recognizing timexes

Systems that perform both recognition and normalization tend to take a rule-based approach to recognition (Mani and Wilson, 2000; Saquete et al., 2002; Schilder, 2004; Negri and Marseglia, 2004). Recognition-only systems, on the other hand, are often based on machine learned classifiers (Hacioglu et al., 2005; Bethard and Martin, 2006), although some such systems do use finite-state methods (Boguraev and Ando, 2005). (Ahn et al., 2005a) show that there can be a benefit to decoupling recognition from normalization, and since our goal is to build a modular, trainable system, we take a machine-learning approach to recognition that is independent of our normalization components.

Generally, machine learned timex recognition systems reduce the task of identifying a timex *phrase* to one of classifying individual *words* by using (some variant of) B-I-O tagging, in which each word as tagged as (B)eginning, (I)nside, or (O)utside a timex phrase. Such a tagging scheme is not inherently sensitive to syntactic constituency and not well-suited to identifying nested timexes (although cf. (Hacioglu et al., 2005)). Considering that syntactic parsers are readily available, and provide reasonable off-the-shelf performance, we have explored using parse information for recognition.

We treat timex recognition as a binary *phrase* classification task: syntactic constituents are classified as timexes or non-timexes. We restrict classification to the following phrase types and lexical categories (based on (Ferro et al., 2004, §5)): NP, ADVP, ADJP, NN, NNP, JJ, CD, RB, and PP.[1] In order to identify candidate phrases and to extract parse-based features, we parse the `TEXT` elements of our documents with the Charniak parser (Charniak, 2000). Because of both parser and annotator errors, only 90.2% of the timexes in the TERN 2004 training data align exactly with a parse, which gives an estimated upper-bound on recall using this method.

We use support vector machines for classification, in particular, the LIBSVM linear kernel implementation (Chang and Lin, 2001). The features we extract include character type patterns, lexical features such as weekday name and numeric year, a context window of two words to the left, and several parse-based features: the phrase type, the phrase head and initial word (and POS tag), and the dependency parent (and corresponding relation) of the head.

4 Component B: Semantic classification

As we mentioned above, timexes may refer to points, durations, or recurrences. While some timexes refer unambiguously to one of these, many timexes are ambiguous between two or even three of

[1] We include PPs despite the TIDES guidelines, which explicitly exclude temporal PPs such as *before Thursday* because of prepositional modifiers such as *around* and *about*.

these. For example, consider the timex *two years* in the following pair of sentences:

(1) Ms. Alsogaray's proposal would require that "substantial progress" on rules for the operational mechanisms be demonstrated a year from now, with a firm decision in *two years*.

(2) Under their deal reached earlier this year, n2k agreed to pay ticketmaster $12 million over *two years*.

In sentence (1), *two years* refers to the point in time two years from when the sentence was written, whereas in sentence (2), it refers to the period of two years during which payments will be made.

In addition to referring to specific points and durations, timexes may refer generically or vaguely, as in following sentences:

(3) Chrysler, Toyota and Honda had their best *October* ever.

(4) *Years ago*, the goaltenders were kind of the guys that couldn't play defense, couldn't play forward, maybe weren't good skaters.

(5) For *years* Republicans have been frustrated by the President's ability to dodge and evade the questions and accusations of investigators.

The timex *October* in sentence (3) refers not to some specific October but generically to the kind of Octobers. The timex *years ago* in sentence (4) refers to some vague point in the past, while the timex *years* in sentence (5) refers to a period of vague length.

Finally, timexes may also refer to recurring times, as in the following sentence:

(6) In the early 1990s, the Defense Advanced Research Projects Agency was spending more than $300 million *a year* on efforts to develop new semiconductor capabilities.

where *a year* refers to the recurring years in the early 1990s.

While the TIMEX2 standard does not explicitly specify semantic classes in its annotations, the semantic classes we distinguish for our normalization system can be easily inferred from the form of the values of the attributes that are annotated, as follows:

Recurrence (recur): SET attribute set to true
Generic or vague duration (gendur): VAL begins with PX or PTX
Duration: VAL begins with P[0-9] or PT[0-9]
Generic or vague point (genpoint): Three possibilities: time-of-day w/o associated date expression (VAL begins with T[0-9]); general reference to past, present, or future (VAL is one of the vague tokens); date expression with unspecified high-order position (i.e., millennium position is X)
Point: Date expression with specified high-order position (may be precise or not—i.e., may include X at other positions—also may be of any granularity, from millennium down to hundredths of a second).

Resolving semantic class ambiguities is a context-dependent task that can be easily factored out of semantic interpretation, reducing the burden on the semantic interpretation rules. The classification task is straightforward: each timex must be classified into one of the five classes described above or into the null class (for timexes that have no VAL). Since the TERN data is not explicitly annotated for semantic class, we use the class definitions above to derive the semantic class of a timex from its VAL attribute.

We again use the LIBSVM linear kernel for classification, with the same features as for recognition. Even though some timexes are unambiguous with respect to semantic class, we train the classifier over all timexes, in the expectation that the contexts of unambiguous timexes will be similar enough to those of ambiguous timexes of the same class to help in classification.

5 Component C: Semantic composition

The semantic composition module uses context-independent, class-specific rules to compute for each timex an underspecified representation—essentially a typed feature structure that depends on the timex's semantic class (with features such as unit and value for durations, or year, month, date, and referential class for points; cf. (Dale and Mazur, 2006)). Because the rules are not responsible for identification or class or direction disambiguation, they are fewer in number and simpler than in other systems (cf. 1000+ for (Negri and Marseglia, 2004)).

Each rule consists of an RE-pattern, which may refer to a small lexicon of names, units, and numeric

words, and is applied using a custom transducer. In total, there are 85 rules, distributed as follows: 11 rules for recurrences; 3 for generic durations; 13 for specific durations; 21 for generic points; 31 for specific points; and 6 class-independent rules to identify information needed to normalize attributes such as MOD, ANCHOR_VAL, and ANCHOR_DIR.

6 Temporal anchors

Some point timexes are fully qualified, while others require a reference time, or temporal anchor, to be fully normalized.[2] There are three ways in which a temporal anchor is chosen for a timex. Some timexes, such as *today* and *three years ago*, are deictic and anchored to the time of speech (in our case, the document timestamp). Other timexes, such as *two months earlier* and *the previous week*, are anaphoric and anchored to a salient time in discourse, in much the same way as an anaphoric pronoun or definite. A timex may also contain its own anchor, as in *two days after May 3*.

Once a referential timex's temporal anchor has been determined, the value of the anchor must be combined with the timex, which may be either an offset or a name-like timexes. Offsets, such as *two months earlier*, provide a unit u, a magnitude m, and optionally, a direction (before or after); the value of an offset is the point (of granularity u) that is m u units from its anchor in the indicated direction. Name-like timexes provide a position in a cycle, such as a day name within a week, and optionally, a direction. The value of a name-like timex is the time point bearing the name within the corresponding cycle of its anchor (or the immediately preceding or succeeding cycle, depending on the direction).

For both offsets and name-like timexes, the direction indication is optional. When no direction indication is given, the appropriate direction must be determined from context, as in this initial sentence from an article from 1998-11-28:

(7) A fundamentalist Muslim lawmaker has vowed to stop a shopping festival planned in *February*, a newspaper reported *Saturday*.

The first timex, *February*, clearly refers to the February following its anchor (the timestamp), while the second timex, *Saturday*, seems to refer to a point preceding its anchor (also the timestamp).

The next two sections describe our methods for temporal anchoring and direction classification.

6.1 Component E: Temporal anchor tracking

Since temporal anchors are not annotated in the TIMEX2 standard, our system uses a simple heuristic method for temporal anchoring. The two most commonly used heuristics are to take the document timestamp or the most recent point timex of fine enough granularity as the temporal anchor of a referential timex. Since we distinguish deictic and anaphoric timexes during semantic composition, we use a combination of these methods: for deictic timexes, the document timestamp is used, and for (some) anaphoric timexes, the most recent point timex, if it is fine-grained enough, is used (otherwise, the document timestamp is used). Because the documents in our corpora are mostly short, we actually treat anaphoric name-like points as deictic and use the most recent timex only for anaphoric offsets.

6.2 Component D: Direction classification

The idea of separating direction classification from the remainder of the normalization task is not new. (Mani and Wilson, 2000) use a heuristic method for this task, while (Ahn et al., 2005b) use a machine learned classifier. In contrast to Ahn et al., who use a set of heuristics to identify ambiguous timexes and train and test only on those, we train our classifier on all point and genpoint timexes and apply it to all point timexes. Genpoint timexes and many point timexes are not ambiguous w.r.t. direction, but we expect that the contexts of unambiguous timexes will be similar enough to those of ambiguous timexes of the same class to help classification.

Direction class is not annotated as part of the TIMEX2 standard. Given a temporal anchor tracking method, though, it is possible to derive imperfect direction class information from the VAL attribute. We use our anchor tracking method to associate each point and genpoint timex with an anchor and then compare the VAL of the timex with that of its anchor to decide what its direction class should be.

We again use the LIBSVM linear kernel for classification. We add two additional sets of features to those used for recognition and semantic classifica-

[2] Our use of the term *temporal anchor* is distinct from the ANCHOR_VAL and ANCHOR_DIR attributes.

tion. The first is inspired by Mani et al., who rely on the tense of neighboring verbs to decide direction class. Since verb tense alone is inherently deictic, it is not sufficient to decide the direction, but we do add both the closest verb (w.r.t. dependency paths) and its POS tag (as well as any verbs directly related to this verb) as features.

The second set of features compares day names, month names, and years to the document timestamp. The comparison determines whether, within a single cycle of the appropriate granularity (week for day-names and year for month-names), the point named by the timex would be before, after, or the same as the point referred to by the timestamp.

7 Details

We trained our system on the training and evaluation data from the TERN 2004 evaluation and the training data from the ACE 2005 evaluation.

Before processing the test data, we parsed it with the Charniak parser (Charniak, 2000). On a 3 Ghz Pentium 4 with 2GB of core memory, parsing took 1 hour and 20 minutes. On the same machine, processing the test data took 368.38 user seconds.

The official results for our revised runs can be found in tables 7 and 7.

Acknowledgments This research was supported by various grants from the Netherlands Organisation for Scientific Research (NWO). Joris van Rantwijk and David Ahn were supported under pro ject number 612.066.302. Maarten de Rijke was supported by NWO under pro ject numbers 017.001.190, 220.80.001, 264.70.050, 354.20.005, 600.065.120, 612.13.001, 612.000.106, 612.066.302, 612.069.006, 640.001.501, and 640.-002.501.

References

David Ahn, Sisay Fissaha Adafre, and Maarten de Rijke. 2005a. Extracting temporal information from open domain text: A comparative exploration. In Roelof van Zwol, editor, *Proceedings of the Fifth Dutch-Belgian Information Retrieval Workshop (DIR'05)*, pages 3–10.

David Ahn, Sisay Fissaha Adafre, and Maarten De Rijke. 2005b. Recognizing and interpreting temporal expressions in open domain texts. In S. Artemov, H. Barringer, A. S. d'Avila Garcez, L. C. Lamb, and J. Woods, editors, *We Will Show Them: Essays in Honour of Dov Gabbay, Vol 1*, pages 31–50. College Publcations.

Steven Bethard and James H. Martin. 2006. Identification of event mentions and their semantic class. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 146–154, Sydney, Australia, July. Association for Computational Linguistics.

Branimir Boguraev and Rie Kubota Ando. 2005. TimeML-compliant text analysis for temporal reasoning. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *IJCAI*, pages 997–1003. Professional Book Center.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st Meeting of NAACL*, pages 132–139.

Robert Dale and Paweł Mazur. 2006. Local semantics in the interpretation of temporal expressions. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 9–16, Sydney, Australia, July. Association for Computational Linguistics.

Lisa Ferro, Laurie Gerber, Inderjeet Mani, and George Wilson, 2004. *TIDES 2003 Standard for the Annotation of Temporal Expressions*. MITRE, April.

Kadri Hacioglu, Ying Chen, and Benjamin Douglas. 2005. Automatic time expression labeling for english and chinese text. In Alexander F. Gelbukh, editor, *CICLing*, volume 3406 of *Lecture Notes in Computer Science*, pages 548–559. Springer.

Inderjeet Mani and George Wilson. 2000. Robust temporal processing of news. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL'2000)*, pages 69–76.

Matteo Negri and Luca Marseglia. 2004. Recognition and normalization of time expressions: ITC-irst at TERN 2004. Technical report, ITC-irst, Trento.

Estela Saquete, Patricio Martínez-Barco, and Rafael Muñoz. 2002. Recognizing and tagging temporal expressions in Spanish. In *Workshop on Annotation Standards for Temporal Information in Natural Language, LREC 2002 (Third International Conference on Language Resources and Evaluation)*, pages 44–51.

Frank Schilder. 2004. Extracting meaning from temporal nouns and temporal prepositions. *ACM Transactions on Asian Language and Information Processing*.

ref	Count				Count (%)					
source	Ent	Detection		Rec	Detection		Rec	Unweighted		
type	Tot	FA	Miss	Err	FA	Miss	Err	Pre	Rec	F
broadca	142	13	36	42	9.2	25.4	29.6	53.8	45.1	49.0
broadca	322	18	55	45	5.6	17.1	14.0	77.9	68.9	73.1
newswir	898	58	210	220	6.5	23.4	24.5	62.7	52.1	56.9
telepho	70	5	22	9	7.1	31.4	12.9	73.6	55.7	63.4
usenet	167	21	33	37	12.6	19.8	22.2	62.6	58.1	60.2
weblog	433	40	103	103	9.2	23.8	23.8	61.4	52.4	56.5
total	2032	155	459	456	7.6	22.6	22.4	64.6	55.0	59.4

Table 1: Count-based scores for revised run

ref	Cost (%)							Unconditioned Cost (%)			
source	Detection		Rec	Value	Value-based			Max	Detection		Rec
type	FA	Miss	Err	(%)	Pre	Rec	F	Value	FA	Miss	Err
broadca	5.9	25.9	21.6	46.6	65.6	52.5	58.3	7.34	0.43	1.90	1.59
broadca	3.0	19.3	9.9	67.8	84.5	70.8	77.1	16.99	0.51	3.27	1.69
newswir	4.4	22.3	16.0	57.3	75.1	61.7	67.8	42.20	1.86	9.40	6.76
telepho	3.1	26.5	6.2	64.2	87.9	67.3	76.2	3.82	0.12	1.01	0.24
usenet	8.4	21.3	11.3	59.0	77.4	67.4	72.0	8.52	0.71	1.82	0.96
weblog	6.2	23.2	15.8	54.8	73.5	61.0	66.7	21.14	1.31	4.91	3.33
total	4.9	22.3	14.6	58.2	76.4	63.1	69.1	100.00	4.95	22.31	14.57

Table 2: Value-based scores for revised run