TimeML-Compliant Text Analysis for Temporal Reasoning

Branimir Boguraev and Rie Kubota Ando
IBM T.J. Watson Research Center, 19 Skyline Drive, Hawthorne, NY 10532, USA
bran@us.ibm.com, rie1@us.ibm.com

Abstract

Reasoning with time[1] needs more than just a list of temporal expressions. TimeML—an emerging standard for temporal annotation as a language capturing properties and relationships among time-denoting expressions and events in text—is a good starting point for bridging the gap between temporal analysis of documents and reasoning with the information derived from them. Hard as TimeML-compliant analysis is, the small size of the only currently available annotated corpus makes it even harder. We address this problem with a hybrid TimeML annotator, which uses cascaded finite-state grammars (for temporal expression analysis, shallow syntactic parsing, and feature generation) together with a machine learning component capable of effectively using large amounts of unannotated data.

1 Temporal Analysis of Documents

Many information extraction tasks limit analysis of time to identifying a narrow class of time expressions, which literally specify a temporal point or an interval. For instance, a recent (2004) ACE task is that of temporal expression recognition and normalisation (TERN; see `http://timex2.mitre.org/tern.html`). It targets absolute date/time specifications (*e.g.* *"June 15th, 1998"*), descriptions of intervals (*"three semesters"*), referential (relative) expressions (*"last week"*), and so forth. A fraction of such expressions may include a relational component (*"the two weeks since the conference"*, *"a month of delays following the disclosure"*), making them event-anchored; however, the majority refer only to what in a more syntactic framework would be considered as a 'temporal adjunct'. The TERN task thus does not address the general question of associating a time stamp with an event.

Deeper document analysis requires awareness of temporal aspects of discourse. Several applications have recently started addressing some issues of time. Document summarisation tackles identification and normalisation of time expressions [Mani & Wilson, 2000], time stamping of event clauses [Filatova and Hovy, 2001], and temporal ordering of events in news [Mani *et al.*, 2003]. Operational question answering (QA) systems can now (under certain conditions) answer *e.g.* *'when'* or *'how long'* questions [Prager *et al.*, 2003].

Beyond manipulation of temporal expressions, advanced content analysis projects are beginning to define operational requirements for, in effect, temporal reasoning. More sophisticated QA, for instance, needs more than just information derived from 'bare' temporal markers [Pustejovsky *et al.*, 2003; Schilder & Habel, 2003]. Intelligence analysis typically handles contradictory information, while looking for mutually corroborating facts; for this, temporal relations within such an information space are essential. Multi-document summarisation crucially requires temporal ordering over events described across the collection.

A temporal reasoner requires a framework capturing the ways in which relationships among entities are described in text, anchored in time, and related to each other. Related are questions of defining a representation that can accommodate components of a temporal structure, and implementing a text analysis process for instantiating such a structure.

This paper describes an effort towards an analytical framework for detailed time information extraction. We sketch the temporal reasoning component which is the ultimate 'client' of the analysis. We motivate our choice of TimeML, an emerging standard for annotation of temporal information in text, as a representational framework; in the process, we highlight TimeML's main features, and characterise a mapping from a TimeML-compliant representation to an isomorphic set of time-points and intervals expected by the reasoner.

We develop a strategy for time analysis of text, a synergistic approach deploying both finite-state (FS) grammars and machine learning techniques. The respective strengths of these technologies are well suited for the challenges of the task: complexity of analysis, and paucity of examples of TimeML-style annotation. A complex cascade of FS grammars targets certain components of TimeML (time expressions, in particular), identifies syntactic clues for marking other components (related to temporal links), and derives features for use by machine learning. The training is on a TimeML annotated corpus; given the small—and thus problematic for training—size of the only (so far) available reference corpus (TimeBank), we incorporate a learning strategy developed

[1]This work was supported by the ARDA NIMD (Novel Intelligence and Massive Data) program PNWD-SW-6059.

to leverage large volumes of unlabeled data.

To our knowledge, this is the first attempt to use the representational principles of TimeML for practical analysis of time. This is also the first use of a TimeML corpus as reference data for implementing temporal analysis.

2 Motivation: Reasoning with Time

We are motivated by developing a useful, and reusable, temporal analysis framework, where 'downstream' applications are enabled to reason and draw inferences over time elements.

A hybrid reasoner [Fikes *et al.*, 2003], to be deployed in intelligence analysis, maintains a directed graph of time points, intervals defined via start and end points, and temporal relations such as BEFORE, AFTER, and EQUAL_POINT. The graph is assumed generated via a mapping process, external to the reasoner, from a (temporal) text analysis. Relations are operationalised, and temporal algebra evaluates instances, draws inference over goals, and broadens a base of inferred assertions on the basis of relational axioms. An example within the reasoner's inferential capability is:

(*find instances of* `?int` *such that* `(during ?int 2003)`).

Reasoning with relations such as *during* (associating an event with an interval), *costarts* (associating two events), instantiated for the example fragment: *"On 9 August Iran accuses the Taliban of taking 9 diplomats and 35 truck drivers hostage in Mazar-e-Sharif. The crisis began with that accusation.*" would infer, on the basis of predicates like:

(during Iran-accuses-Taliban-take-hostages August-9-1998)
(costarts Iran-accuses-Taliban-take-hostages Iran-Taliban-Crisis)

that the answer to the question *"When did the Iranian-Taliban crisis begin?"* is *"August 9, 1998"*.

Details of this inferential process need not concern us here. We gloss over issues like enumerating the range of temporal relations and axioms, describing the reasoner's model of events (*e.g. Iran-accuses-Taliban-take-hostages*), and elaborating its notion of 'a point in time' (subsuming both literal expressions and event specifications). Operationally, a separate component maps temporal analysis results to a suitably neutral, and expressive, ontological representation of time (DAML-Time [Hobbs *et al.*, 2002]). This allows for a representation hospitable to first-order logic inference formalism—like the one assumed in Hobbs *et al.*—to be kept separate from surface text analysis: much like the traditional separation along the syntax-semantics interface.

We start from the belief that the representation for the reasoner is derivable from a TimeML-compliant text analysis.[2] TimeML is a proposal for annotating time information;*e.g.* the first example sentence above would be marked up as:

[2]We are not alone: work on temporal reasoning from formal inference point of view reaches a similar conclusion: "... the [TimeML] annotation scheme itself, due to its closer tie to surface texts, can be used as the first pass in the syntax-semantics interface of a temporal resolution framework such as ours. The more complex representation, suitable for more sophisticated reasoning, can then be obtained by translating from the annotations." [Han & Lavie, 2004].

```
<SIGNAL sid="s1"> On </SIGNAL> <TIMEX3 tid="t1" type="DATE" value=
"1998-08-09"> 9 August </TIMEX3> Iran <EVENT eid= "e1" class= "I_ACTION">
accuses </EVENT> the Taliban of  <EVENT eid="e4" class="OCCURRENCE"> taking
</EVENT> 9 diplomats and 35 truck drivers hostage in Mazar-e-Sharif. The <EVENT
eid="e8" class="OCCURRENCE"> crisis </EVENT> <EVENT eid="e12" class=
"ASPECTUAL"> began </EVENT> <SIGNAL sid="s2" type="DATE" mod="START"> with
</SIGNAL> that <EVENT eid="e16" class="I_ACTION"> accusation </EVENT>.
<MKINSTANCE eiId="ei1" evId="e1"/> <MKINSTANCE eiId="ei2" evId="e8"/>
<MKINSTANCE eiId="ei3" evId="e12"/> <MKINSTANCE eiId="ei4" evId="e16"/>
<TLINK eiId="ei1" relToTime="t1" relType="IS_INCLUDED"/>
<TLINK eiId="ei4" relToEIId="ei1" relType="IDENTITY"/>
<ALINK eiId="ei2" relToEIId="ei4" relType="INITIATES"/>
```

TimeML is described in Section 3. Essentially, it promotes explicit representation and typing of time expressions and events, and an equally explicit mechanism for linking these with temporal links, using a vocabulary of temporal relations.

In addition to in-line mark-up, explicit links are marked. Event instance identifiers, ei1, ei2, and ei4 refer to, respectively, the accusation in the first sentence, the crisis, and the reference to *"that accusation"* in the second sentence. The relType attributes on the link descriptions define temporal relationships between event instances and time expressions; in this particular example, an IDENTITY link encodes the co-referentiality between the event instances (mentions) in the two sentences of the *accusation* event of the earlier example.

It is the combination of event descriptors, their anchoring to time points, and the semantics of relational links, which enable the derivation of *during* and *costarts* associations that the reasoner understands.

3 TimeML: a Mark-up Language for Time

Most content analysis applications to date do not explicitly incorporate temporal reasoning, and their needs can be met by analysis of simple time expressions (dates, intervals, *etc*). This is largely the motivation for TERN's TIMEX2 tag; at the same time it explains why TIMEX2 is inadequate for supporting the representational requirements outlined earlier.[3]

TimeML aims at capturing the richness of time information in documents. It marks up more than just temporal expressions, and focuses on ways of systematically anchoring event predicates to a time denoting expressions, and on ordering such event expressions relative to each other.

TimeML derives higher expressiveness from explicitly separating representation of temporal expressions from that of events. Time analysis is distributed across four component structures: TIMEX3, SIGNAL, EVENT, and LINK; all are rendered as tags, with attributes [Saurí *et al.*, 2004].[4]

[3]For a notable extension to TIMEX2, see [Gaizauskas & Setzer, 2002]. An attempt to codify some relational information linking the TIMEX with an event, it is still limited, both in terms of scope (only links with certain syntactic shape can be captured) and representational power (it is hard to separate an event mention from possibly multiple event instances); see [Pustejovsky *et al.*, 2003].

[4]Additionally, a MKINSTANCE tag embodies the difference between event *tokens* and event *instances*: for example, the analysis of *"Max taught on Monday and Tuesday"* requires two different instances to be created for a *teaching* EVENT. Even if typically there is a one-to-one mapping between an EVENT and an instance, the language requires that a realisation of that event is created.

TIMEX3 extends[5] the TIMEX2 [Ferro, 2001] attributes: it captures temporal expressions (commonly categorised as DATE, TIME, DURATION), both literal and intensionally specified. SIGNAL tags are (typically) function words indicative of relationships between temporal objects: temporal prepositions (*for*, *during*, *etc*.) or temporal connectives (*before*, *while*). EVENT, in TimeML nomenclature, is a cover term for situations that happen or occur; these can be punctual, or last for a period of time. TimeML posits a refined typology of events [Pustejovsky *et al.*, 2003]. All classes of event expressions—tensed verbs, stative adjectives and other modifiers, event nominals—are marked up with suitable attributes on the EVENT tag. Finally, the LINK tag is used to encode a variety of relations that exist between the temporal elements in a document, as well as to establish an explicit ordering of events. Three subtypes to the LINK tag are used to represent strict temporal relationships between events or between an event and a time (TLINK), subordination between two events or an event and a signal (SLINK), and aspectual relationship between an aspectual event and its argument (ALINK).

TimeML's richer component set, in-line mark-up of temporal primitives, and non-consuming tags for temporal relations across arbitrarily long text spans, make it highly compatible with the current paradigm of annotation-based encapsulation of document analysis.

4 TimeML and Temporal Analysis

TimeML's annotation-based representation facilitates integration of time analysis with the analysis of other syntactic and/or discourse phenomena; it also naturally supports exploitation of larger contextual effects by the temporal parser proper (see 4.4) . This is a crucial observation, given that the prominently attractive characteristic of TimeML—its intrinsic richness of expression—makes it challenging for analysis.

There are two broad categories of problems for developing an automated TimeML analyser: of substance and of infrastructure. Substantive issues include normalising time expressions to a canonical representation (TIMEX3's value attribute), identifying a broad range of events (*e.g.* event nominals and predicative adjectives acting as event specifiers), linking time-denoting expressions (typically a TIMEX3 and an EVENT), and typing of those LINKs.

The infrastructure problems—small size and less than consistent mark-up of the TimeBank corpus—are due to the fact that this, first, version is largely a side product of a small number of annotators trying out TimeML's expressive capabilities. TimeBank is thus intended as a reference, and not for training. Our hybrid approach to temporal parsing, combining finite-state (FS) recognition with machine learning from sparse data (4.2), is largely motivated by this nature of TimeBank.

4.1 The TimeBank corpus

TimeBank has only 186 documents (68.5K words). If we held out 10% of the corpus as test data, we have barely over 60K words for training. Below we show counts of (EVENT-TIMEX3) TLINK[6] and EVENT types [Saurí *et al.*, 2004]. TLINK examples are particularly sparse; the data also shows highly uneven distribution of examples of different types.

In comparison, the Penn TreeBank corpus for part-of-speech tagging contains >1M words (> 16 times larger than TIMEBANK); the CoNLL'03 named entity chunking training set (at `http://cnts.uia.ac.be/conll2003/ner/`) has over 200K words with 23K examples (15 times more than TLINK examples) over just 4 name classes (compared to the 13 TLINK classes defined by TimeML). TERN's training set—almost 800 documents/300K words—is considered to be somewhat sparse, with over 8K TIMEX examples.

TLINK type	# occurrences	EVENT type	# occurrences
IS_INCLUDED	866	OCCURRENCE	4,452
DURING	146	STATE	1,181
ENDS	102	REPORTING	1,010
SIMULTANEOUS	69	I_ACTION	668
ENDED_BY	52	I_STATE	586
AFTER	41	ASPECTUAL	295
BEGINS	37	PERCEPTION	51
BEFORE	35		
INCLUDES	29		
BEGUN_BY	27		
IAFTER	5		
IDENTITY	5		
IBEFORE	1		
Total :	1,451	Total :	8,243

4.2 Analytical strategy

Minimally, the reasoner would require that the analytical framework supports time stamping and temporal ordering of events; thus we target the analysis tasks of finding TIMEX3's, assigning canonical values, marking and typing EVENTs, and associating (some of them) with TIMEX3 tags.

TIMEX3 expressions are naturally amenable to FS description. FS devices can also encode some larger context for time analysis (temporal connectives for marking putative events, clause boundaries for scoping possible event-time pairs, *etc*; see 4.4). To complement such analysis, a machine learning approach can cast the problem of marking EVENTs as chunking. Recently, [Ando, 2004] has developed a framework for exploiting large amounts of unannotated corpora in supervised learning for chunking. In such a framework, mid-to-high-level syntactic parsing—typically derived by FS cascades—can produce rich features for classifiers.

Thus, we combine FS grammars for temporal expressions, embedded in a general purpose shallow parser, with machine learning trained with TimeBank and unannotated corpora.

4.3 FS-based parser for temporal expressions

Viewing TIMEX3 analysis as an information extraction task, a cascade of finite-state grammars with broad coverage (compiled down to a single TIMEX3 automaton with 500 states and over 16000 transitions) targets abstract temporal entities such

[5]TIMEX2 and TIMEX3 differ substantially in their treatment of event anchoring and sets of times.

[6]In all of our experiments we exclude TIMEX3 markup in metadata; the TLINK counts only reflect links to temporal expressions in the body of documents.

as UNIT, POINT, PERIOD, RELATION, *etc*; these may be further decomposed and typed into *e.g.* MONTH, DAY, YEAR (for a UNIT); or INTERVAL or DURATION (for a PERIOD).

Fine-grained analysis of temporal expressions, instantiating attributes like `granularity`, `cardinality`, `ref_direction`, and so forth, is crucially required for normalising a TIMEX3: representing *"the last five years"* as illustrated below facilitates the derivation of a value for the TIMEX3 value attribute.

```
[timex : [relative      : true ]
         [ref_direction : past ]
         [cardinality   : 5 ]
         [granularity   : year ] ]
```

Such analysis amounts to a parse tree under the TIMEX3. (Not shown above is additional information, anchoring the expression into the larger discourse and informing other normalisation processes which emit the full complement of TIMEX3 attributes—type, temporalFunction, anchorTimeID, *etc*). TimeBank does not contain such fine-grained mark-up: the grammars thus perform an additional 'discovery' task, for which no training data currently exists, but which is essential for discourse-level post-processing, handling *e.g.* ambiguous and/or underspecified time expressions or the relationship between document-internal and document-external temporal properties (such as 'document creation time').

4.4 Shallow parsing for feature generation

In principle, substantial discourse analysis can be carried out from a shallow syntactic base, and derived by means of FS cascading [Kennedy & Boguraev, 1996]. Our grammars interleave shallow parsing with named entity extraction. They specify temporal expressions in terms of *linguistic* units, as opposed to simply lexical cues (as many temporal taggers to date do). This point cannot be over-emphasised. One of the complex problems for TimeML analysis is that of event identification. A temporal tagger, if narrowly focused on time expressions only (cf. [Schilder & Habel, 2003]), offers no clues to what events are there in the text. In contrast, a temporal parser aware of the syntax of a time phrase like *"during the long and ultimately unsuccessful war in Afghanistan"* is very close to knowing—from configurational properties of a prepositional phrase—that the nominal argument (*"war"*) of the temporal preposition (*"during"*) is an event nominal.

Ultimately, syntactic analysis beyond TimeML components is used to derive features for the classifiers tasked with finding EVENTs and LINKs (Section 5).

Feature generation typically relies on a mix of lexical properties and some configurational syntactic information (depending on the complexity of the task). Our scheme additionally needs some semantic typing, knowledge of boundaries of longer syntactic units (typically a variety of clauses), and some grammatical function. An example (simplified) of the FS cascade output is:

```
[Snt [svoClause
  [tAdjunct In [NP [timex3 the 1988 period timex3] NP] tAdjunct],
  [SUB [NP the company NP] SUB]
  [VG [GrmEventOccurrence earned grmEventOccurrence] VG]
  [OBJ [NP [Money $20.6 million Money] NP] OBJ] svoClause] ... Snt]
```

Most of the above is self-explanatory, but we emphasise a few key points. The analysis captures the mix of syntactic chunks, semantic categories, and TimeML components used for feature generation. It maintains local TIMEX3 analysis; the time expression is inside of a larger clause boundary, with internal grammatical function identification for some of the event predicates. The specifics of mapping configurational information into feature vectors is described in Section 5.

4.5 Machine learning for TimeML components

TimeML parsing is thus a bifurcated process of TimeML components recognition: TIMEX3's are marked by FS grammars; SIGNALs, EVENTs and LINKs are identified by classification models derived from analysis of both TimeBank and large unannotated corpora. *Features* for these models are derived from common strategies for exploiting local context, as well as from mining the results—both mark-up and configurational—from the FS grammar cascading, as illustrated in the previous section. (More details on feature generation follow in Section 5 below.)

Classifiers and feature vectors

The classification framework we adopt for this work is based on a principle of *empirical risk minimization*. In particular, we use a *linear classifier*, which makes classification decisions by thresholding inner products of feature vectors and weight vectors. It learns weight vectors by minimizing classification errors (*empirical risk*) on annotated training data.

For our experiments (Section 6), we use the *Robust Risk Minimization* (RRM) classifier [Zhang *et al.*, 2002], which has been shown useful for a number of text analysis tasks such as syntactic chunking, named entity chunking, and part-of-speech tagging.

In marked contrast to generative models, where assumptions about features are tightly coupled with algorithms, RRM—as is the case with discriminative analysis—enjoys clear separation of feature representation from the underlying algorithms for training and classification. This facilitates experimentation with different feature representations, since the separation between these and the algorithms which manipulate them does not require change in algorithms. We show how choice of features affects performance in Section 6.

Word profiling for exploitation of unannotated corpora

In general, classification learning requires substantial amount of labeled data for training—considerably more than what TimeBank offers (cf. 4.1). This characteristic of size is potentially a limiting factor in supervised learning approaches. We, however, seek to improve performance by exploiting unannotated corpora, with their natural advantages of size and availability. We use a *word profiling* technique, developed specially for exploiting a large unannotated corpus for tagging/chunking tasks [Ando, 2004]. Word profiling identifies, and extracts, word-characteristic information from unannotated corpora; it does this, in essence, by collecting and compressing feature frequencies from the corpus.

Word profiling turns co-occurrence counts of words and features (*e.g.* 'next word', 'head of subject', *etc*) into new feature vectors. For instance, observing that *"extinction"* and *"explosion"* are often used as syntactic subject to *"occur"*, and that *"earthquakes" "happen"*, helps to predict that *"explosion"*, *"extinction"*, and *"earthquake"* all function like event nominals. Below (6.1) we demonstrate the effectiveness of word profiling, specifically for EVENT recognition.

5 Implementation

To use classifiers, one needs to design *feature vector representation* for the objects to be classified. This entails selection of some predictive attributes of the objects (in effect promoting these to the status of *features*) and definition of mappings between vector dimensions and those attributes (*feature mapping*). In this section we describe the essence of our feature design for EVENT and TLINK recognition.[7]

5.1 EVENT recognition

Similarly to named entity chunking, we cast the EVENT recognition task as a problem of sequential labeling of tokens by encoding chunk information into token tags. For a given *class*, this generates three tags: E:*class* (the last, end, token of a chunk denoting a mention of *class* type), I:*class* (a token inside of a chunk), and O (any token outside of any target chunk). The example sequence below indicates that the two tokens *"very bad"* are spanned by an event-state annotation.

$\cdots$ *another*/O *very*/I:event-state *bad*/E:event-state *week*/O $\cdots$

In this way, the EVENT chunking task becomes a $(2k+1)$-way classification of tokens where k is the number of EVENT types; this is followed by a Viterbi-style decoding. (We use the same scheme for SIGNAL recognition.)

The feature representation used for EVENT extraction experiments mimics the one developed for a comparative study of entity recognition with word profiling [Ando, 2004]. The features we extract are:

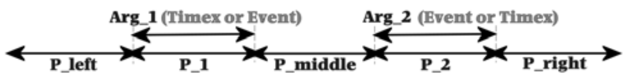
- ○ token, capitalization, part-of-speech (POS) in 3-token window;
- ○ bi-grams of adjacent words in 5-token window;
- ○ words in the same syntactic chunk;
- ○ head words in 3-chunk window;
- ○ word uni- and bi-grams based on subject-verb-object and preposition-noun constructions;
- ○ syntactic chunk types (noun or verb group chunks only);
- ○ token tags in 2-token window to the left;
- ○ tri-grams of POS, capitalization, and word ending;
- ○ tri-grams of POS, capitalization, and left tag.

5.2 TLINK recognition

TLINK is a relation between events and time expressions which can link two EVENTs, two TIMEX3's, or an EVENT and a TIMEX3. Presently (see 4.2) we focus on TLINKs between events and time expressions.

As a relational link, TLINK does not naturally fit the tagging abstraction for a chunking problem, outlined above. Instead, we formulate a classification task as follows. After posting EVENT and TIMEX3 annotations (by the event classifier and the FS temporal parser, respectively), for each pairing between an EVENT and a TIMEX3, we ask whether it is a certain type of TLINK. This defines a $(\ell+1)$-way classification problem, where ℓ is the number of TLINK types (BEFORE, AFTER, *etc*; Section 4.1). The adjustment term '$+1$' is for the *negative* class (not-a-temporal-link).

The relation-extraction nature of the task of posting TLINKs requires a different feature representation, capable of encoding the syntactic function of the relation arguments (EVENTs and TIMEX3's), and some of the larger context of their mentions. To that end, we consider the following five *partitions* (defined in terms of tokens): spans of arguments (P_1 or P_2); two tokens to the left/right of the left/right argument (P_left/P_right); and the tokens between the arguments (P_middle). From each partition, we extract tokens and parts-of-speech as features.



We also consider *segments* (syntactic constructions derived by FS analysis: 'when-clause', 'subject', *etc*) in certain relationship to partitions: contained in P_1, P_2, or P_middle; covering P_1 (or P_2) but not overlapping with P_2 (or P_1); occurring to the left of P_1 (or the right of P_2); or covering both P_1 and P_2. We use uni- and bi-grams of types of these segments as features.

In this feature representation, segments play a crucial role by capturing the syntactic functions of EVENTs and TIMEX3's, as well as the syntactic relations between them.

Thus in the example analysis on p. 4, `svoClause` is the smallest segment containing both an EVENT and a TIMEX3, indicative of a direct syntactic relation between the two. In the next example, the TIMEX3 and EVENT chunks are contained in different clauses (a `thatClause` and a `svoClause`, respectively), which structurally prohibits a TLINK relation between the two.

```
[Snt
 Analysts have complained
 [thatClause that [timex3 third-quarter timex3] corporate earnings
    have n't been very good thatClause]
 [svoClause , but the effect [event hit event] ... svoClause] Snt]
```

Thus our feature representation is capable of capturing this information via the types of the segments that contain each of EVENT and TIMEX3 without overlapping.

6 Experiments

We present here performance results on EVENT and TLINK recognition only. This is largely because the primary focus of this paper is to report on how effective our analytical strategy is in leveraging the reference nature of the small TimeBank corpus for training classifiers for TimeML. Of these, SIGNAL was briefly mentioned earlier (see footnote 7), and TIMEX3 recognition, driven by FS grammars, belongs to a different paper. Since this is the first attempt to build a TimeML-compliant analyser (cf. Section 1), there are no comparable results in the literature.

The results (micro-averaged F-measure) reflect experiments with different settings, against the TimeBank corpus, and produced by 5-fold cross validation.

[7]We do not discuss SIGNAL recognition here, as the `signal` tag itself contributes nothing to EVENT or TLINK recognition, beyond what is captured by a lexical feature over the temporal connective.

6.1 EVENT recognition

It should be clear, by looking at the example analysis (p. 4), how local information and syntactic environment both contribute to the feature generation process. Figure 1 shows performance results with and without word profiling for exploiting an unannotated corpus. For word profiling, we extracted feature co-occurrence counts from 40M words of 1991 *Wall Street Journal*. The proposed event chunks are counted as correct only when both the chunk boundaries and event types are correct. While word profiling improves performance, 64.0% F-measure is lower than typical performance of, for instance, named entity chunking. On the other hand, when we train the EVENT classifiers without typing, we obtain 80.3% F-measure. This is indicative of the inherent complexity of the EVENT typing task.

features	with typing	w/o typing
basic	61.3	78.6
basic + word-profiling	64.0 (+2.7)	80.3 (+1.7)

Figure 1: Event extraction results, with/without typing. Parentheses show contribution of word profiling, over using basic features only.

6.2 TLINK recognition

In this experimental setting, we only consider the pairings of EVENT and TIMEX3 which appear within a certain distance in the same sentences.[8]

For comparison, we implement the following simple baseline method. Considering the text sequence of EVENTs and TIMEX3's, only 'close' pairs of potential arguments are coupled with TLINKS; EVENT e and TIMEX3 t are close if and only if e is the closest EVENT to t and t is the closest TIMEX3 to e. For all other pairings, no temporal relation is posted. Depending on the 'with-'/'without-typing' setting, the baseline method either types the TLINK as the most populous class in TimeBank, IS_INCLUDED, or simply marks it as 'it exists'.

Results are shown in Figure 2. Clearly, the detection of temporal relations between events and time expressions requires more than simply coupling the closest pairs within a sentence (as the baseline does). It is also clear that the baseline method performs poorly, especially for pairings over relatively long distances. For instance, it produces 34.9% when we consider the pairings within 64 tokens without typing. In the same setting, our method produces 74.8% in F-measure, significantly outperforming the baseline.

distance (# of TLINKs)	features	with typing	w/o typing
distance ≤ 64 tokens (1370 TLINKs)	baseline	21.8	34.9
	basic	52.1	74.1
	basic+FS	53.1 (+1.0)	74.8 (+0.7)
distance ≤ 16 tokens (1269 TLINKs)	baseline	38.7	61.3
	basic	52.8	75.8
	basic+FS	54.3 (+1.5)	76.5 (+0.7)
distance ≤ 4 tokens (789 TLINKs)	baseline	49.8	76.1
	basic	57.0	80.1
	basic+FS	58.8 (+1.8)	81.8 (+1.7)

Figure 2: TLINK extraction results, with/without typing. Parentheses show contribution of grammar-derived features, over using basic ones only. Baseline posts TLINKs over 'close' EVENT/TIMEX3 pairs.

We compare performance in two types of feature representation: 'basic' and 'basic+FS grammar', which reflect the without- and with-segment-type information obtained by the grammar analysis, respectively. As the positive delta's show, configurational syntactic information can be exploited beneficially by our process. Focusing on within-4-tokens pairings, we achieve 81.8% F-measure without typing of TLINKs, and 58.8% with typing. (The task without typing is a binary classification to detect whether the pairing has a TLINK relation or not, regardless of the type.) As the figure shows, the task becomes harder when we consider longer distance pairings. Within a 64 token distance, we obtain figures of 74.8% and 53.1%, without and with typing respectively.

While we are moderately successful in detecting the *existence* of temporal relations, the noticeable differences in performance between the task settings with and without typing indicate that we are not as successful in distinguishing one type from another. In particular, the relatively low performance of TLINK typing highlights the difficulty in distinguishing between DURING and IS_INCLUDED.

The guidelines (and common sense analysis) suggest that IS_INCLUDED type should be assigned if the time point or duration of EVENT *is included* in the duration of the associated TIMEX3. DURING, on the other hand, should be assigned as a type if some relation represented by the EVENT *holds during* the duration of the TIMEX3. We note that for this particular typing problem, the subtle distinctions are hard even for human annotators: the TimeBank corpus displays a number of occasions where inconsistent tagging is evident.

7 Conclusion

TimeML is a significant development in time analysis, as it captures detailed information, anchored in eventuality and linguistic structure, and shown to be crucial inferential and reasoning tasks. In addition to defining annotation guidelines, the TimeML effort notably created the first reference corpus illustrative of expressiveness of the language.

Unfortunately, the small size of the TimeBank corpus prevents its straightforward use as a training resource, a problem further exacerbated by the inherent complexity of TimeML-compliant analysis. And yet, for reasoning engines to function, TimeML analysers need to be built.

[Mani *et al.*, 2004] discuss some pioneering work in linking events with times, and ordering events, indicative of productive strategies for posting (some) TLINK information. However, the nature of these efforts is such that differences in premises, representation, and focus make a direct performance comparison impossible. Furthermore, the work pre-dates TimeML, and cannot be conveniently mapped to TimeBank data; this, in effect, precludes a quantitative comparison with our work.

In a first systematic attempt at TimeML-compliant analysis,

[8]To evaluate the TLINK classifier alone, we use the EVENT and TIMEX3 annotations in TimeBank.

and leveraging the TimeBank corpus, we have developed a strategy which synergistically blends finite-state analysis over linguistic annotations with a state-of-the-art machine learning technique. Particularly effective are: aggressive analysis, by complex grammars, of both TimeML components and syntactic structure; coupled with a learning algorithm capable of training over unannotated data, in addition to exploiting arbitrarily small amounts of labeled data. While work remains (notably refining the TLINK recogniser, targeting other types of LINKs, and enhancing EVENT recognition with external lexical resources), this is a significant step in instantiating a deeper time analysis, capable of satisfying the needs of reasoning engines.

References

[Ando, 2004] R.K. Ando. Exploiting unannotated corpora for tagging and chunking. In *Proceedings of ACL-04*.

[Ferro, 2001] L. Ferro. TIDES: Instruction manual for the annotation of temporal expressions. MTR 01W0000046V01, The MITRE Corporation, 2001.

[Fikes *et al.*, 2003] R. Fikes, J. Jenkins, and G. Frank. JTP: A system architecture and component library for hybrid reasoning. TR KSL-03-01, Stanford University, 2003.

[Filatova and Hovy, 2001] E. Filatova & E. Hovy. Assigning time-stamps to event-clauses. In *Proceedings of the 10th Conference of the EACL*, Toulouse, France, 2001.

[Gaizauskas & Setzer, 2002] R. Gaizauskas and A. Setzer, editors. *Annotation Standards for Temporal Information in NL*, Las Palmas, Spain, 2002.

[Han & Lavie, 2004] B. Han and A. Lavie. Framework for resolution of time in natural language. *TALIP Special Issue, Spatial and Temporal Information Processing*, 2004.

[Hobbs *et al.*, 2002] J.R. Hobbs, G. Ferguson, J. Allen, P. Hayes, and A. Pease. A DAML ontology of time, 2002.

[Kennedy & Boguraev, 1996] C. Kennedy & B. Boguraev. Anaphora for everyone: Pronominal anaphora resolution without a parser. In *Proceedings of COLING-96*, Copenhagen, DK, 1996.

[Mani & Wilson, 2000] I. Mani and G. Wilson. Robust temporal processing of news. In *Proceedings of the 38th Annual Meeting of the ACL*, Hong Kong, 2000.

[Mani *et al.*, 2003] I. Mani, B. Schiffman, and J. Zhang. Inferring temporal ordering of events in news. In *Proceedings of ACL-41 (HLT-NAACL)*, Edmonton, Canada, 2003.

[Mani *et al.*, 2004] I. Mani, J. Pustejovsky, and B .Sundheim. Introduction: special issue on temporal information processing. *ACM Transactions Asian Language Information Processing*, 3(1):1–10, 2004.

[Prager *et al.*, 2003] J. Prager, J. Chu-Carroll, E. Brown, and C. Czuba. Question answering using predictive annotation. In *Advances in Question Answering*, 2003.

[Pustejovsky *et al.*, 2003] J. Pustejovsky, J. Castaño, R. Ingria, R. Saurí, R. Gaizauskas, A. Setzer, G. Katz, and D. Radev. TimeML: Robust specification of event and temporal expressions in text. In *AAAI Spring Symposium on New Directions in Question-Answering*, pages 28–34, Stanford, CA, 2003.

[Saurí *et al.*, 2004] R. Saurí, J. Littman, R. Gaizauskas, A. Setzer, and J. Pustejovsky. TimeML annotation guidelines, Version 1.1, TERQAS Workshop, 2004.

[Schilder & Habel, 2003] F. Schilder and C. Habel. Temporal information extraction for temporal QA. In *AAAI Spring Symposium on New Directions in Question-Answering*, pages 35–44, Stanford, CA, 2003.

[Zhang *et al.*, 2002] T. Zhang, F. Damerau, and D.E. Johnson. Text chunking based on a generalization of Winnow. *Journal of Machine Learning Research*, 2:615–637, 2002.