RC23455 (W0411-204) November 29, 2004
Computer Science

IBM Research Report

TimeML-Compliant Analysis of Text Documents

Branimir K. Boguraev, Rie K. Ando
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

TimeML-Compliant Analysis of Text Documents

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

Abstract

Reasoning with temporal information[1] requires a representation of time considerably more involved than just a list of temporal expressions—which typically define the extent of current time extraction efforts. TimeML is an emerging standard for temporal annotation, defining a language for expressing properties and relationships among time-denoting expressions and events in free text. This paper takes the position that TimeML is a good starting point for bridging the gap between temporal analysis of documents and reasoning with information derived from these documents. TimeML-compliant analysis is hard; and the task is made even harder by the small size of the only annotated corpus available to date. To address this, and related, challenges, we have developed and implemented a hybrid TimeML annotator, which uses cascaded finite-state grammars (for temporal expresison analysis, shallow syntactic parsing, and feature generation) together with a machine learning component capable of effectively using large amounts of unannotated data. We motivate our mixed strategy; this is work in progress, and we report interim results on the first effort to use the TIMEBANK corpus for building an operational TimeML analyser.

[1]This work was supported by the Advanced Research and Development Activity under the Novel Intelligence and Massive Data (NIMD) program PNWD-SW-6059.

Contents

1 Temporal Analysis of Documents

Broadly speaking, practical content analysis of documents relies on a variety of 'gisting' approaches, offering surrogate views into what a document is about. Thus numerous NLP technologies and applications are concerned with identifying high information quotient-bearing text fragments. Typical of such approaches are, for instance, efforts to extract mentions of named entities and broader semantic categories of concepts: in isolation, chained, or linked in relational structures. These trends can be observed in the definition of community-wide efforts like the Message Understanding Conferences (MUC)[2] and the Automatic Content Extraction (ACE) evaluations.[3]

Documents, of course, are about much more than entities and concepts alone. Typically, they focus on connected narratives, which describe a broad range of relationships among entities. Indeed, the evolution of content analysis tasks, from MUC to ACE, ultimately targeting some relation identification, reflects this.

One of the characteristic properties—if not *the* defining property—of such relationships is that all of them have a temporal component. Still, the only extent to which some of the MUC or ACE tasks address time analysis is to look at a relatively narrow class of time expressions, namely those literally specifying a temporal point or an interval. For instance, one of the most recent ACE tasks is defined as temporal expression recognition and normalisation (TERN). It addresses the identification of, for instance, explicit (absolute) date and time specifications (*e.g. "June 15th, 1998"*), descriptions of intervals and periods (*"three semesters"*), time points described by reference (relative expressions: *"last week"*), and so forth [TERN, 2004]. A fraction of such expressions may include a relational component (*"the two weeks since the conference"*, *"a month of delays following the disclosure"*), reflecting their manifestation as event-anchored expressions; however, the majority refer only to what in a larger analysis space would be considered as a 'temporal adjunct'. The TERN task thus does not address the general question of associating a time stamp with a relation.

Broader document analysis requires awareness of temporal aspects in the discourse. Therefore, different applications of content analysis have recently started addressing some issues of time. In particular, work in automatic document summarization has addressed questions like identification and normalisation of time expressions [Mani and Wilson, 2000], time

[2] See `http://www.itl.nist.gov/iaui/894.02/related_projects/muc/main.html`.

[3] See `http://www.nist.gov/speech/tests/ace/index.htm`.

stamping of event clauses [Filatova and Hovy, 2001], and temporal ordering of events in news [Mani et al., 2003]. In the context of question answering (QA), operational systems can now produce (literal) answers to *e.g. 'when'* or *'how long'* questions (assuming a document contains a factual statement with an explicit time-related label such as TIME, DATE, or DURATION) [Prager et al., 2003].

Going beyond manipulation of temporal expressions alone, a number of advanced content analysis projects are beginning to define operational requirements for some form of temporal reasoning. More sophisticated question answering, for instance, needs to concern itself with more than utilising just information derived from 'bare' temporal markers (as those illustrated above): see, for instance, [AAAI/QA, 2003], and more specifically, [Pustejovsky et al., 2003], [Schilder and Habel, 2003]. Intelligence analysis scenarios typically involve sifting through contradictory information, while looking for strands of mutually corroborating facts; temporal relations within such an information space are essential for such an operation. Multi-document summarisation applications crucially require temporal ordering over events described in more than a single document.

For a reasoner to be made aware of relevant information in documents, a framework is required for capturing the ways in which relationships among entities are described in text, anchored in time, and related to each other. This, in turn, raises the complementary questions of defining a representation rich and flexible enough to accommodate components of a temporal structure, and implementing a text analysis process capable of instantiating such a structure.

Our work falls in this space, as it is driven by the requirements of an on-going project developing a support infrastructure for an intelligence analyst's workstation. Extraction of temporal information, temporal reasoning, and derivation and manipulation of timelines are crucial components of such an infrastructure.

This paper describes a continuing effort to develop an analytical framework for extraction of detailed time information. We briefly outline the temporal reasoning component (Section 2) which is the ultimate 'client' of the analysis. We motivate our choice of a representational framework for time; in the process, we highlight the main features of TimeML, an emerging standard for the annotation of temporal information in documents (Section 3). As part of the motivation, we sketch a mapping process which derives, from a TimeML-compliant representation, an isomorphic set of timepoints and intervals: the grist to a temporal reasoner's mill.

The core of the paper elaborates on a strategy for temporal analysis of

document text, which implements a synergistic approach deploying both finite-state grammars and machine learning techniques (Section 4). It turns out that the respective strengths of these technologies play out particularly well, given the complexities of the task, the requirements of the framework, and the relative paucity of examples of TimeML-style annotation.A complex cascade of finite-state grammars targets certain components of TimeML (such as time expressions and temporal connectives; see Section 3) directly, identifies syntactic context rich with clues for marking other components (such as event verbs and event-denoting nominals), and maps the source text into a set of carefully engineered features for use by learning machines; these are trained with a TimeML annotated corpus, and deploy a novel learning strategy specially developed to leverage large volumes of unlabeled data—thus circumventing the mismatch between the complexity of TimeML analysis and the smallness of the only (so far) available reference TimeML corpus (hereafter TIMEBANK; see Section 4.1).

To our knowledge, this is the first attempt to use the representational principles of TimeML for analysis of time in an operational system. This is also the first time that a TimeML corpus is being actively used as reference data for developing a temporal analysis process. We highlight some of the challenges arising from the complexity of the task and the paucity of annotated data, and show how our synergistic approach meets these challenges. Ours is work in progress. We thus describe our current experimental setup, and report interim, but promising, performance resuls on analysing TimeML components.

2 Motivation: Reasoning with Time

The motivation for our work derives, in large part, from considerations of utility and reusability of a temporal analysis framework. In essence, we want to enable 'downstream' applications to reason and draw inferences over time elements.

For example, under development in Stanford's Knowledge Systems Laboratory is a hybrid reasoner [Fikes et al., 2003], to be deployed in intelligence analysis scenarios. The reasoner maintains a directed graph of time points, which is based on temporal relations such as BEFORE, AFTER, and EQUAL_POINT; it also represents intervals using their starting and ending points. Temporal relations are operationalised, and temporal algebra facilitates evaluation over instances, draws inference over instances of goals, and broadens a base of inferred assertions on the basis of relational axioms.

An operation within the reasoner's inferential capability would be exemplified by the following directive:

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

For a given text, the reasoner would assume a mapping has happened, converting the temporal aspects of the text analysis into a graph with relations such as (among others) *during* (associating an event with a time point), *costarts* (associating two events), *etc.* These would be instantiated, for instance, in the example below.

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. On 2 November Iran concludes the Zolfaghar-2 military exercise peacefully, ending the crisis between the two sides. On 5 September Iran states that it has the right under international law to strike the Taliban after Iranian media sources report that the Taliban have killed 5 Iranian diplomats.

Given such a graph, on the basis of predicates like:

(during Iran-accuses-Taliban-of-taking-hostages August-9-1998)

(costarts Iran-accuses-Taliban-of-taking-hostages Iranian-Taliban-Crisis)

---

the reasoner would, for instance, infer that the answer to the question "*When did the Iranian-Taliban crisis begin?*" is "*August 9, 1998*". (Note that this kind of question answering, targeting implicit relationships rather than explicit facts, also crucially relies on the ability to link two event specifications, in the first and second sentences respectively, as co-referential; this is, however, outside of the scope of this paper.)

The details of this inferential process need not concern us here. We gloss over representational issues like enumerating the range of temporal relations and axioms (*e.g. during* and *costarts*), describing the reasoner's model of events (*e.g. Iran-accuses-Taliban-of-taking-hostages*), and elaborating its notion of 'a point in time' (which subsumes both literal temporal expressions, suitably normalised, and event specifications). Operationally, a separate component is responsible for mapping the results of the temporal analysis proper to a suitably neutral, and expressive, ontological representation

of time (such as, for instance, DAML-Time [Hobbs et al., 2002]). This reflects the notion that a representation hospitable to, say, a first-order logic inference formalism—like the one assumed in Hobbs *et al.*—is necessarily removed from the surface text analysis: much along the lines separating the traditional syntax-semantics interface.

This paper focuses on the initial analysis aspect in such an interface. Central to our argument is the belief that the particular representation assumed by the reasoner is derivable from a TimeML analysis of the same text. We are not alone in this: interestingly, work approaching the problem of temporal reasoning from the formal inference end, also 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 translationg from the annotations." [Han and Lavie, 2004].

By way of illustration, as well as informal introduction to TimeML, the fragment below, for the first two sentences of the text, shows an analysis.

```
<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>
.
<makeinstance eiid="ei1" eventId="e1"/>
<makeinstance eiid="ei2" eventId="e8"/>
<makeinstance eiid="ei3" eventId="e12"/>
<makeinstance eiid="ei4" eventId="e16"/>
<tlink eventInstanceId="ei1" relatedToTime="t1"
       relType="IS\_INCLUDED"/>
<tlink eventInstanceId="ei4" relatedToEventInstance="ei1"
       relType="IDENTITY"/>
<alink eventInstanceId="ei2" relatedToEventInstance="ei4"
       relType="INITIATES"/>
```

The 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. Notice the relType attributes on the link

descriptions. These define, typically, a temporal relationship between an event instance and a time expression; in this particular example, an IDENTITY link encodes the co-referentiality between the event instances (mentions) in the two sentences (cf. the *Iran-accuses-Taliban-of-taking-hostages* event of the earlier example).

It is the combination of event descriptors, their anchoring to time points (*e.g.* t1, namely *"9 August"*), and the semantics of relational links, which makes it possible to derive *during* and *costarts* associations that the reasoner understands from the particular combination of IS_INCLUDED, IDENTITY and INITIATES relational labels in the TimeML analysis.

For simplicity, the illustration is incomplete (some tags and attributes are omitted); more details on TimeML are given in section 3 below. The essential points of this analysis to note here are: explicit, and separate, representation and typing of time expressions and events, and an equally explicit mechanism for linking these with temporal links which incorporate a vocabulary of temporal relations.

3 Schemes for Temporal Annotation

The NLP community is still in relatively early stages of establishing uniform guidelines and practices for representing time information. As illustrated by the earlier (see Section 1) enumeration of content analysis applications which do carry out some analysis of time, their needs can largely be met by the identification and normalisation of simple time expressions (dates, intervals, reference points, and so forth). Consequently, broadly accepted schemes for temporal analysis such as TIDES [Ferro, 2001]—most recently used in the TERN evaluation [TERN, 2004]—focus on defining guidelines for folding time information within a tag for temporal expressions (TIMEX[4]); this is clearly inadequate for supporting the representational requirements outlined in the previous section.

A notable extension of the TIMEX representational scheme is proposed by [Gaizauskas and Setzer, 2002], who make an attempt to incorporate some relational information between a time expression and an event within a broader definition of a TIMEX tag. Still, this has its limitations, both in terms of scope (only temporal links expressed as certain syntactic forms can be captured) and representational power (it is hard to separate an event mention from possibly multiple event instances); see [Pustejovsky et al., 2003]. Thus even this scheme would fall short in encoding the richer kind of

[4]For largely historical reasons, the label "TIMEX2" is used.

analysis that a temporal reasoning system like the one discussed in Section 2 above would need in order to maintain knowledge about time points, intervals, and temporal relations between time expressions and events.

3.1 TimeML: a language for time

A broad community effort, TERQAS (Temporal and Event Recognition for QA Systems)[5], over the last 24 months has undertaken the design of a special purpose representation language for events and temporal expressions. TimeML, aims at being able to capture the richness of time information in documents. In particular, TimeML goes beyond specification of a tagging scheme for temporal expressions only (*e.g.* the TIMEX2 tag of TIDES and TERN), and focuses, among other things, on ways of systematically anchoring event predicates to a broad range of temporally denotating expressions, and on ordering such event expressions (relative to each other). The language provides for delayed evaluation of contextually underspecified, or partially determined, time expressions (such as "*last year*" and "*two months before*"). What follows is a brief sketch of TimeML's characteristic features; [Pustejovsky et al., 2003] offer more details.

TimeML derives its larger expressive power by means of explicitly separating the representation of temporal expressions from that of events. Additionally, it allows for anchoring, or ordering, of dependencies that may exist in text. The reresentation makes use of four component structures: TIMEX3, SIGNAL, EVENT, and LINK; all four are rendered as tags, with attributes, annotating text spans.

TIMEX3 extends[6] the TIDES TIMEX2 [Ferro, 2001] attributes; it is taken to denote temporal expressions (subsuming common notions like DATE, TIME, DURATION), as well as intensionally specified expressions like the examples above, handled by the definition of suitable temporal functions. SIGNAL is a tag for (typically) function words which indicate how temporal objects are to be related to one another; examples here include temporal prepositions (like *for*, *during*, *at*) or temporal connectives (*before*, *after*, *while*). EVENT, as used in TimeML nomenclature, is a cover term for situations that happen or

[5] See `http://www.timeml.org/terqas/index.html`.

[6] Substantial differences between TIMEX2 and TIMEX3 are in their treatment of event anchoring and sets of times. Relational time expressions (*two days before departure*) are a single TIMEX2, but map to a collection of related TIMEX3, SIGNAL and EVENT tags. Sets of times (*every week*) get different analysis. This has effect both on the boundaries of annotation spans, and on attributes of the covering annotations (tags).

occur; these can be punctual, or last for a period of time.[7]

TimeML posits a refined ontology 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 properties 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).

Without going into specific detail, the flavour of a TimeML representation can be further conveyed by showing the analysis, and tagging, of the sentence "*The terrorists convened two days before the attack*".

```
The terrorists
<EVENT eid="e1" class="OCCURRENCE" tense="PAST" aspect="PERFECTIVE">
convened
</EVENT>
<TIMEX3 tid="t1" type="DURATION" value="P2D" temporalFunction="false">
two days
</TIMEX3>
<SIGNAL sid="s1">before</SIGNAL>
the
<EVENT eid="e2" class="OCCURRENCE" tense="NONE" aspect="NONE">
attack
</EVENT>
<MAKEINSTANCE eiid="ei1" eventID="e1"/>
<MAKEINSTANCE eiid="ei2" eventID="e2"/>
<TLINK eventInstance="ei1" signalId="s1" relatedToEvent="ei2"
       relType="BEFORE" magnitude="t1"/>
```

For complete description of the complete set of attributes on TimeML tags, see [Saurí et al., 2004]. Here we only reiterate one of the fundamental differences between this representation and annotation schemes like, for instance, TERN, which typically would mark the larger phrase, "*two days before the attack*", as a single TIMEX2, collectively encoding the temporal specification, the event and the link between the two as attributes of the same tag (cf. p. 3).

[7] We observe that TimeML 'events' can be interpreted as ontological events, as well as lexicalisations of relations, as targeted by *e.g.* ACE- or KDD-like information extraction efforts. This observation underlies the process we alluded to earlier (p. 4): namely the mapping of TimeML EVENTs into ontologically valid relational structures, as mandated by the reasoner's model of the world (Section 2).

By positing a richer set of components (made even more expressive by custom attributes), by inlining the markup of temporal primitives, and by defining non-consuming tags for specifying (a range of different) relations among those across arbitrarily long text spans, TimeML makes it possible to express complex facts about the temporal properties of discourse in a way which fits the current paradigm of annotation-based encapsulation of document analysis.

4 TimeML and Temporal Analysis

TimeML is, by design, robust enough to capture the rich system of interrelationships underlying the sequencing of events in a document. In Section 2 above we outlined how a temporal analysis of a discourse, couched in TimeML terms, can be mapped into a form encapsulating a variety of temporal links, and usable by a reasoning component. This suggests that TimeML would be a suitable transport mechanism in an analytical framework spanning the syntax-semantics interface assumed by reasoners. Together with the other factors discussed earlier, it motivates our choice of TimeML as a representational scheme targeted by our analysis.

TimeML conforms to the established practices for posting analysis results as annotations over text fragments; this facilitates integration of time analysis with the analysis of other syntactic and / or discourse phenomena; as we will see below (Section 4.3.2), it also naturally supports the exploitation of larger contextual effects by the time analysis proper. This is a crucial observation, because the prominently attractive characteristic of TimeML—its intrinsic richness of expression—makes it, at the same time, hard to deal with, in its complexity.

There are two broad categories of problems for developing an automated TimeML analyser: of substance and of infrastructure. Of particular substantive difficulty are issues like normalising time expressions to a canonical representation (the value attribute on TIMEX3 tags), identification of a broad range of events (including, for instance, event nominals and predicative adjectives acting as event specifiers), linking time-denoting expressions (including links between a TIMEX3 and an EVENT), and typing of those LINKs. These issues remain valid even after we narrow down, for practical purposes (see Section 4.2 below), the range of phenomena tackled by an initial implementation.

The infrastructure problems, on the other hand, derive from the fact that TimeML is a set of broad annotation guidelines for human annota-

tors, which is still not fully finalised. One particular consequence of this is that the only available reference corpus of TimeML-annotated documents is small, and not (yet) fully consistent internally.

It is the combined complexity of the annotation task and paucity of pre-annotated data, that make motivate hybrid approach of finite-state grammars coupled with machine learning, which we will focus on in Section 4.2 below. First, we look at some of the characteristics of the TIMEBANK corpus, from the point of view of a machine learning component.

4.1 The TimeBank corpus

The TERQAS workshop committed to exercising the annotation standard on a reference corpus, TIMEBANK.[8] While this is broadly illustrative of how the annotation guidelines should be applied, the corpus is currently too small to be adequately usable as a language resource from which a learning machine could be fully trained.

For instance, standard corpora for (the relatively simple task of) part-of-speech tagging typically consist of over 1M words. The CONLL-2003 named entity chunking training set[9] consists of over 200K words—and that excludes the development and test sets. TERN's TIMEX2 training set consists of almost 800 documents and 300K words; even so, with just over 8000 examples of temporal expressions, it is considered to be somewhat sparse.

By contrast, TIMEBANK has only 186 documents, with a total of 68.5K words. If we held out 10% of the corpus as test data, we have only barely over 60K words for training.

As a further illustration of its sparseness, consider the typing of TLINKs between events and temporal expressions. [Saurí et al., 2004] define 13 link types; TIMEBANK contains 1,451 instances of TLINKs, with an uneven distribution of their types (866 IS_INCLUDED, 146 DURING, compared with 5 each for IAFTER, IDENTITY, and 1 IBEFORE; see Section 5.2 below for some additional description). To put this into perspective, CoNLL named entity data offers 33K examples for just 4 classes.

4.2 Analytical strategy

Clearly, the question of how optimally to leverage the reference corpus data for a realistic analysis engine is crucial to an effort like ours. As we have seen, there simply is not enough training data for a supervised learning

[8] See `http://nrrc.mitre.org/NRRC/Docs_Data/MPQA_04/approval_time.htm`

[9] See `http://cnts.uia.ac.be/conll2003/ner/`

approach, by itself. At the same time, an alternative, grammar-based approach, also faces certain difficulties: in effect, we would need a full syntactic parser coupled with custom discourse processing engine to derive some of the links and types among TimeML components—at the very least, this would be a time-consuming effort, considering the detail of lexical support which would be required for complete identification and typing of events.

It is reasonable to try and put some boundaries on the problem. For our practical and immediate needs, the analytical framework needs to be capable of identifying TIMEX3 expressions, assigning canonical values to them, marking and typing EVENTs, and associating (some of these, as appropriate) with TIMEX3 tags. This is the minimal set of operations required for time-stamping and temporal ordering of events and relations in a document, still producing rich enough analysis base to feed reasoners.

Within such a reduced problem space, it is easier to see the advantages of each of the alternatives above. The nature of temporal (TIMEX3) expressions is well suited to formalising them by means of finite-state (FS) grammars. FS devices can also encode some of the larger context of relevance to time analysis (*e.g.* temporal connectives for marking putative event lexemes, clause boundaries for delimiting the scope of a possible event-time pairing, and so forth; see Section 4.3 below). Using suitable classification methods, a machine learning approach can address the problem of marking EVENTs as a sequential labeling problem. Going back to the question posed at the beginning of this section, some recent work in machine learning holds promise by developing a framework in which large amounts of unlabeled data can be profitably exploited by semi-supervised learning techniques, thus counteracting the paucity of labeled data [Ando, 2004a]. In such a framework, it has been shown that mid-to-high-level syntactic chunking—typically derived by means of FS grammar cascades—produces rich features for classifiers.[10]

Our strategy thus exercises a synergistic approach which makes use of a FS parser for temporal expressions embedded in a general purpose shallow parser and a machine learning component trained with the TIMEBANK reference corpus and unlabeled data.

[10]Finally, viewing TLINKs as relational links over two (strongly typed) arguments allows us to address the time-stamping problem by an extension of our methodology which aggressively uses large amounts of unlabeled data; this is work in its early stages, and we will return to it in a separate paper.

4.3 Finite-state components for time analysis

Finite state technology performs two different functions in our overall strategy. First, it is the primary device implementing a parser for temporal expressions. Second, TIMEX3 analysis is itself embedded in a separate, FS-based, shallow syntactic parser. This provides a backdrop against which larger contextual effects with implications for temporal analysis are discovered; the syntactic analysis (with suitable modifications and enhancements to handle temporal constructs) is itself then used as a source of features (see Section 4.4.1 below) for the machine learning component.

We use a flexible FST system within an annotations-based pipelined architecture for document analysis [Boguraev and Neff, 2003, Neff et al., 2003].

4.3.1 A parser for temporal expressions

The use of FS techniques is naturally motivated by the observation that temporal expressions follow a set of broadly regular patterns, amenable to grammar-based description. The syntax of temporal expressions is such that, given an expressive formalism for writing patterns over linguistic annotations, it is possible to cover a broad range of open-ended expression types. Viewing TIMEX3 analysis as an information extraction task, we have developed a system of finite-state grammars with broad coverage[11] for abstract temporal entities, such as UNIT, POINT, SPAN, PERIOD, RELATION, and so forth; these may be further decomposed and typed into *e.g.* MONTH, DAY, YEAR (for a UNIT); or INTERVAL or DURATION (for a PERIOD).

The fine-grained decomposition of a temporal expression into components like INTERVAL, GRANULARITY, CARDINALITY, REF_DIRECTION is crucially required by the process of normalising the TIMEX3 into a canonical form: it is the representation of *"the last five years"* as

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

which facilitates the derivation of "5PY" as the string value of the TIMEX3 VALUE attribute.

[11] With all the subgrammars for temporal entities compiled, TIMEX3 analysis is captured by a single automaton with over 300 states and close to 3000 transitions

In essence, the normalisation process expects as input constellations of finer-grained analysis which amounts to a requirement for a parse tree under the TIMEX3 annotation span (Such a tree would contain additional information, not shown in the simplified example above, which 'anchors' the time expression into the larger document discourse and subsequently informs other normalisation processes which emit the full complement of TIMEX3 attributes—such as TYPE, TEMPORALFUNCTION, ANCHORTIMEID, BEGINPOINT/ENDPOINT, and so forth). This parse is not only straightforwardly achieved by cascading FS grammars; given the nature of the TIMEBANK corpus, which does not contain such fine-grained detail, training for TIMEX3 attributes would be seriously challenged. A grammar-based analysis thus naturally situates a TIMEX3 parse into a temporal discourse analysis[12] which deals with *e.g.* ambiguous and/or underspecified time expressions, or the relationship between document-internal and document-external temporal properties (such as 'document creation time').

4.3.2 Feature generation by shallow parsing

In principle, large amount of discourse analysis can be carried out starting from a shallow syntactic base, and derived by means of FS cascading (as argued, for instance, in [Kennedy and Boguraev, 1996]). Furthermore, our analytical framework already makes extensive use of FS-based information extraction capability targeting a broad range of ontologically relevant semantic categories. Thus we interleave general purpose shallow parsing with named entity extraction, which makes it possible to realise the two crucial points to our strategy: patterns over temporal expressions can be specified in terms of *linguistic* units, as opposed to simply lexical cues (as many temporal taggers to date do); and both intermediate and final levels of syntactic analysis can be directly exploited to generate features for the machine learning component (see section 5.3).

The point of specifying patterns over *linguistic* units cannot be overemphasised. One of the big issues in TIMEML analysis, as we have seen, is that of event identification. A temporal tagger, if narrowly focused on time expressions only (see, for instance, [Schilder and Habel, 2003]), offers no clues to what events are there in the text. A temporal parser, on the other hand, capable of syntactic decomposition of a temporal phrase like *"during the long and ultimately unsuccessful war in Afghanistan"* is very close to knowing—by virtue of interpreting the syntactic constraints underlying

[12] This is carried out by post-processing, outside of the grammars.

a prepositional phrase—that the noun phrase head *"war"* which is the argument of the temporal preposition *"during"* is an event nominal.

This kind of information is easily captured within a parsing framework. Additionally, given that EVENTs and LINKs are ultimately posted by a machine learning component, the parser need not commit to *e.g.* event identification and typing. It can gather clues, and formulate hypotheses; and it can then make these available to an appropriate classifier.

Smooth integration of temporal and syntactic analysis is thus achieved by coordinating the temporal grammars with a shallow syntactic parser, also realised [Boguraev, 2000] as a cascade of finite-state devices. Using, in addition, a mechanism for accessing external resources, it is also possible to query an authority file about the event-denoting status of certain lexical items (such as nominal expressions) in key syntactic positions (such as heads of noun phrases). We expect to find reference material like [Levin, 1993] and [Wood, 1967] particularly useful here. This facilitates the implementation of a temporal parser which, in effect, deposits three types of TIMEML tags into the document analysis stream: TIMEX3, SIGNAL, and EVENT. The output of the temporal parser for *"For nearly forty years, the United States has said categorically it would not tolerate totalitarian rule in its own backyard."* is exemplified below.

```
[svoClause
 [tAdjunct
  [GrmSignal For GrmSignal]
  [NP
   [ngHead
    [timex nearly [cardinal [cardUnit forty cardUnit] cardinal]
                  [pUnit years pUnit] timex] ngHead] NP] tAdjunct]
 ,
 [SUB [NP the [ngHead [Country United States Country] ngHead] NP] SUB]
 [VG has
     [vgHead [GrmEventReport said GrmEventReport] vgHead]
     categorically VG] svoClause]
[svoClause
 [SUB [NP [ngHead it ngHead] NP] SUB]
 [VG would not
     [vgHead [GrmEventIntAct tolerate GrmEventIntAct] vgHead] VG]
 [OBJ
  [NPP
   [NP totalitarian [ngHead rule ngHead] NP]
   [PP in
       [PossNP [NPS its NPS]
            [NP own [ngHead backyard ngHead] NP] PossNP] PP] NPP] OBJ]
. svoClause]
```

While most of the above is self-explanatory, it is worth pointing out a few key points concerning the analysis. It captures the mix of syntactic chunks, semantic categoris, and TimeML components; this facilitates the extraction of a broad range of features for the subsequent classifiers (in particular, for EVENT and SIGNAL; see Section 5.3.1). It maintains the local TIMEX3 analysis; the time expression is situated inside of a larger clause boundary, with internal grammatical function identification for some of the event predicates, which will be used to derive features for the classifier tasked with posting of TLINKs (see Section 5.3.3). It also distinguishes between the different status the TimeML components identified by the parser have with respect to the classifiers: TIMEX3 tags are part of the final analysis, SIGNALS, EVENTS and EVENT types are only putatively suggested (as indicated by the composite labels GRMSIGNAL, GRMEVENTREPORT), and later used as features by the classifiers.

4.4 Machine learning for TimeML components

Thus TimeML parsing is, for us, a process of TimeML components recognition, where SIGNALS, EVENTS and LINKS (but not TIMEX3's) are identified on the basis of classification models derived from analysis of both TIMEBANK and large unnanotated corpora. *Features* for these models are derived from common strategies for exploiting locality of context, as well as from mining the results—both markup and configurational—from the FS grammar cascading, as illustrated in the previous section. (More details on feature generation follow in Section 5.3 below.)

4.4.1 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 the annotated training data. There are good reasons to use linear classifiers; an especially good one is that they allow for easy experimentation with various types of features, without making any model assumptions.

To use linear 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*). For instance, for our EVENT recognition, tokens need be classified as belonging to the inside or the outside of an EVENT chunk (EVENT references are not necessarily single words; see Section 5.1.2). Examples of typical features are strings of the current and neighboring words.

Linear classification models enjoy a clear separation of feature representation from the underlying algorithms for training and classification. This is in marked contrast to generative models (*e.g.* hidden Markov models for part-of-speech tagging) where assumptions about features are tightly coupled with algorithms. Thus using linear classifiers further facilitates experimentation with different feature systems, by maintaining separation between these and the algorithms which manipulate them, and thus not mandating algorithms change.

We will show how choice of features affects performance in Section 5.

4.4.2 Word profiling for exploitation of unannotated corpora

In general, classification learning requires a substantial amount of labeled data for training. As we have shown already (Section 4.1), the TIMEBANK corpus—the only existing reference corpus for TimeML-compliant analysis—is very small compared with standard annotated corpora for this sort of tasks.

This characteristic of size is potentially a limiting factor in machine learning approaches. We, however, seek to improve performance by exploiting unannotated corpora, which have the natural advantages of being sizable, and freely available. We use a *word profiling* technique, developed specially for the purposes of exploiting a large unannotated corpus for tagging/chunking tasks [Ando, 2004a, Ando, 2004b]. Word profiling identifies, and extracts, information that characterizes words from unannotated corpora; it does this, in essence, by collecting and compressing feature frequencies from the unannotated corpus, a process which maps the commonly used feature vectors to frequency-encoded context vectors.

More precisely, word profiling counts co-occurrences of words and features (such as 'next word', 'head of subject', *etc*), and use the count data as part of new feature vectors. For instance, the observations, in an unannotated corpus, that nouns like *extinction* and *"explosion"* are often used as the syntactic subject to *"occur"*, and that *"earthquakes" "happen"*, help to predict that *"explosion"*, *"extinction"*, and *"earthquake"* all function like event nominals. We will demonstrate the effectiveness of word profiling, specifically on the EVENT recognition task, in the next section.

5 Experiments

Within the constraints for our analysis, as defined in Section 4.2, we focus on the identification of a reduced set of TimeML components. Consequently, we report here performance results on EVENT, SIGNAL, and TLINK recognition (TLINKs are the only LINK type currently targeted by our TimeML annotator).

5.1 Implementation

5.1.1 RRM classifier

For our experiments, we use the *Robust Risk Minimization* (RRM) classifier [Zhang et al., 2002]. RRM has been shown to be useful for a number of text analysis tasks such as syntactic chunking [Zhang et al., 2002], named entity chunking [Florian et al., 2003, Zhang and Johnson, 2003, Florian et al., 2004], and part-of-speech tagging [Ando, 2004a].

5.1.2 EVENT and SIGNAL recognition

As is often done for 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 *class*, this gives rise to three tags: E:*class* (the last, end, token of an chunk denoting a mention of *class* type), I:*class* (a token inside of a chunk), and O (for any token outside of any target chunks). For instance, the sequence

$$\cdots \quad \textit{another}\,/\,\mathsf{O} \quad \textit{very}\,/\,\mathsf{I{:}event\text{-}state} \quad \textit{bad}\,/\,\mathsf{E{:}event\text{-}state} \quad \textit{week}\,/\,\mathsf{O} \quad \cdots$$

indicates that the two tokens *"very bad"* are spanned by an event-state annotation.

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

5.1.3 TLINK recognition

TLINK is a relation between events and time expressions.[13] A TLINK can thus link two EVENTs, or two TIMEX3 expressions, or an EVENT and a

[13] Canonically, an event also denotes a time expression.

TIMEX3. As described in Section 4.2 earlier, for practical purposes in this work we focus on TLINKs between events and time expressions.

Given that a TLINK is, in essence, a relational link, it does not naturally fall into the tagging abstraction underlying the chunking problem, as outlined in Section 5.1.2 above. We formulate a classification task as follows. After EVENT and TIMEX3 are annotated (respectively by the event classifier, as described in Section 5.1.2, and the FS temporal parser, Section 4.3.1), for each pairing between an EVENT and a TIMEX3, we ask whether this pair is a certain type of TLINK. That is, we have a $(\ell + 1)$-way classification problem where ℓ is the number of TLINK types such as BEFORE and AFTER (the following section offers more details on TLINK types). The adjustment term '$+1$' is for the *negative* class that indicates the pair does not have any kind of temporarl link relation.

5.2 Data statistics

Some counts characteristic of the distribution of TimeML components in the TIMEBANK corpus were given earlier, in Section 4.1. Overall, in 186 documents and 68,448 tokens, there are 2,148 SIGNALs, 8,243 EVENTs, and 1,451 TLINKs (we do not report the number of TIMEX3's, as that data is not used for training of the machine learning component).

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

Figure 1: Distribution of TLINK and EVENT types in TIMEBANK corpus.

Figure 1 shows breakdown of the distribution of the two components which require typing. For description of the semantics of the type labels, see [Saurí et al., 2004].

5.3 Performance results

The performance results reported in this section reflect experiments with different settings. The results are reported against the TIMEBANK corpus, and were produced by 5-fold cross validation.

5.3.1 EVENT

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

- token, capitalization, 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.

(a) with typing

features	P	R	F
primitive	63.1	59.7	61.3
primitive+word-profiling	65.6 (+2.5)	62.6 (+2.9)	64.0 (+2.7)

(b) w/o typing

features	P	R	F
primitive	78.5	78.8	78.6
primitive+word-profiling	80.7 (+2.2)	79.9 (+1.1)	80.3 (+1.7)

Figure 2: Event extraction performance results, with and without typing. Micro-averaged precision, recall, and F-measure. Numbers in parentheses show positive contribution of word profiling, over using primitive features only. 5-fold cross validation.

It should be clear, by looking at the example analysis (p. 16), how local information and syntactic environment both contribute to the feature generation process.

Figure 2 shows performance results in the settings with and without word profiling for exploiting an unannotated corpus. For word profiling, we extracted feature co-occurrence counts from 40 million words of 1991 *Wall Street Jounal* articles. In Figure 2 (a), 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 classifiers for EVENT without typing, we obtain 80.3% F-measure (Figure 2 (b)). This is indicative of the inherent complexity of the EVENT typing task.

The performance results on each event type is shown in Figure 3.

event types	P	R	F
REPORTING	87.8	90.0	88.9
OCCURRENCE	67.8	70.4	69.1
PERCEPTION	67.3	68.6	67.9
ASPECTUAL	64.8	51.2	57.2
I_STATE	59.0	52.9	55.8
I_ACTION	49.1	43.3	46.0
STATE	40.6	27.9	33.1

Figure 3: Event extraction performance results, with breakdown of EVENT types.

We note here that the TIMEBANK corpus occasionally displays inconsistencies (*e.g.* the same verb, in similar contexts, is marked either as OCCURRENCE or as I_ACTION), omissions (*e.g.* a verb will not be marked as an event, even when it is clearly functioning as such), or confusion (*e.g.* MONEY amounts are occasionally marked as OCCURRENCE events).

5.3.2 SIGNAL

For SIGNAL experiments, we use the same features as above; additionally, we use (putative) SIGNAL annotations produced by FS grammars analysis as features.The SIGNAL chunks proposed by grammars are encoded into token tags using IEO scheme (as described in Section 5.1.2), and those tags are used as features of token instances. Figure 4 shows that the additional features produced by the grammars in this manner, indeed, improve per-

formance. The set of grammars for SIGNAL extraction used for these experiments is still incomplete. We expect to obtain further performance improvements by expanding it.

features	P	R	F
primitive	65.0	63.5	64.2
primitive+grammar	65.0 (0.0)	65.2 (+1.7)	65.1 (+0.9)

Figure 4: Signal extraction performance results. Precision, recall, and F-measure. Numbers in parentheses show positive contribution of syntactic analysis, over using primitive features only. 5-fold cross validation.

The word profiling technique is not so effective for SIGNAL extraction. This is not surprising, given that the technique is designed to counteract the sparseness of open-class words, whereas the majority (if not all) of SIGNAL occurrences are typically with function words (e.g., *"after"*, *"not"*, *etc*).

5.3.3 TLINK

The TLINK extraction task is similar to that of relation extraction, where an EVENT and a TIMEX3 are arguments to the relation. Thus it requires a different feature representation. First, we consider the following five types of partitions:

- EVENT chunks,
- TIMEX3 chunks,
- two tokens to the left of the left argument (EVENT or TIMEX3),
- two tokens to the right of the right argument (TIMEX3 or EVENT),
- the tokens between EVENT and TIMEX3,

Next, for each partition, we extract the following features:

- tokens,
- parts-of-speech,
- types of *segments* that are contained in the partition, or that contain the partition. If the contained partition covers an argument, overlap the other argument is prohibited. If it does not cover an argument, then overlap with any partition covering an argument is prohibited. Segments are syntactic constructions obtained by the FST analysis—for instance, 'when-clause', 'that-clause', 'subject', 'object', and so forth (see Section 4.3.1, and example on p.16),

- bi-grams of the above segment types.

We represent the syntactic relations between EVENT and TIMEX3 by:

- the type of the shortest segment containing both the EVENT and TIMEX3 chunks,
- bi-grams that combine the types of segments containing the EVENT chunk and TIMEX3 chunk, respectively, and
- tri-grams that combine the types of segments containing the EVENT chunk, the TIMEX3 chunk, and the partition between the EVENT and TIMEX3 chunks, respectively.

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 an EVENT and a TIMEX3.

Thus in the example analysis below (somewhat simplified for legibility), `svoClause` is the smallest segment that contains both EVENT and TIMEX3, and is indicative of a direct syntactic relation between the TIMEX3 and EVENT.

```
[Snt
  [svoClause
    In [timex3 the 1988 period timex3],
    [SUB the company SUB]
    [event earned event]
    [OBJ $20.6 million OBJ] svoClause] ... Snt]
```

In the next example, the TIMEX3 and EVENT chunks are contained in different clauses (a 'that-clause' and a 'svo-clause', respectively), which structurally prohibits a TLINK relation between the two. Our feature representation adequately captures this information via the types of the segments that contain each of EVENT and TIMEX3 without overlapping.

```
[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]
```

with typing, within 64 tokens			
features	P	R	F
baseline	14.6	42.7	21.8
primitive	55.6	49.1	52.1
primitive+grammar	56.8 (+1.2)	50.0 (+0.9)	53.1 (+1.0)
w / o typing, within 64 tokens			
baseline	23.4	68.4	34.9
primitive	74.7	73.6	74.1
primitive+grammar	74.7 (0.0)	74.9 (+1.3)	74.8 (+0.7)
with typing, within 16 tokens			
baseline	33.7	45.5	38.7
primitive	55.8	50.0	52.8
primitive+grammar	57.6 (+1.8)	51.4 (+1.4)	54.3 (+1.5)
w / o typing, within 16 tokens			
baseline	53.3	72.0	61.3
primitive	75.4	76.2	75.8
primitive+grammar	76.5 (+1.1)	76.6 (+0.4)	76.5 (+0.7)
with typing, within 4 tokens			
baseline	45.5	55.0	49.8
primitive	56.9	57.1	57.0
primitive+grammar	59.1 (+2.2)	58.5 (+1.4)	58.8 (+1.8)
w / o typing, within 4 tokens			
baseline	69.5	84.0	76.1
primitive	77.4	83.0	80.1
primitive+grammar	79.5 (+2.1)	84.3 (+1.3)	81.8 (+1.7)

Figure 5: TLINK extraction performance results with and without typing. Micro-averaged precision, recall, and F-measure. Numbers in parentheses show positive contribution of grammar-derived features, over using primitive features only. 5-fold cross validation. Baseline marks TLINKs over 'close' pairs of EVENTs and TIMEX3's.

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

We implement the following baseline method. Considering the text sequence of EVENTs and TIMEX3's, only 'close' pairs of potential arguments are coupled with TLINKs; EVENT e_j and TIMEX3 t_k are close if and only if e_j is the closest EVENT to t_k and t_k is the closest TIMEX3 to e_j. For all other pairings, no temporal relation is posted. Depending on the 'with-' or 'without-' typing setting, we either type the TLINK as the most populous class in TIMEBANK, IS_INCLUDED, or simply mark it as 'it exists'.

The F-measure results are shown in Figure 5. It is clear that the detection of temporal relations between events and time expressions requires more than simply coupling the closest pairs within a sentence. It is also clear that the baseline method performs poorly especially for the pairings in relatively long distances. For instance, it produces 34.9% in F-measure 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.

We compare performance in two types of feature representation: 'primitive' and 'primitive+grammar', which reflect the without- and with-the-segment-type information obtained by the grammar analysis, respectively. As the numbers in parentheses (and, ultimately, the F-measure differences) demonstrate, configurational syntactic information can be exploited beneficially by our process. When we focus on the pairings within 4 tokens, we achieve 81.8% F-measure without the typing of TLINK, 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 seen from the figure, the task becomes harder when we consider pairings across longer distances. 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. Certain cases of wrong analysis by our system are just that: wrong, arguably due to insufficient training data. We note, however, 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.

Finally, in Figure 6, we show the performance on the three types on which our system achieved the best F-measure. We will reiterate here that 70% is very respectable, given the complexity of the TIMEML annotation task and the paucity of training data.

within 64 tokens				
TLINK types	occ#	P	R	F
IS_INCLUDED	796	60.3	73.6	66.3
BEGINS	31	81.3	41.9	55.3
ENDED_BY	49	52.2	24.5	33.3
within 16 tokens				
IS_INCLUDED	778	61.4	74.9	67.5
BEGINS	30	82.4	46.7	59.6
ENDED_BY	45	57.9	24.4	34.4
within 4 tokens				
IS_INCLUDED	509	63.3	80.2	70.7
BEGINS	11	71.4	45.5	55.6
ENDED_BY	25	73.3	44.0	55.0

Figure 6: TLINK extraction performance results for the three 'easiest' categories. Features are primitive + grammar. 5-fold cross validation.

6 Conclusion

TimeML is a significant development in the area of time analysis of text, as it explicitly addresses the question of providing sufficient detail, anchored in eventuality and linguistic structure, of information shown to be crucial to a broad range of inferential and reasoning tasks. In addition to defining a set of annotation guidelines, the TimeML effort notably created the first reference corpus illustrative of the kinds of analysis the language is designed to capture.

Unfortunately, the size of the TIMEBANK corpus makes it hard for it to be used effectively by machine learning approaches alone; this problem is further exacerbated by the complexity of the phenomena at the focus of TimeML-compliant analysis. And yet, for reasoning engines to function, TimeML analysers need to be built.

In this paper we report on the first systematic attempt to build such an analyser; furthermore, we seek to fully utilise the availablity of a reference corpus. In order to be able to exploit the data in that corpus, we have proposed, and developed, a strategy which synergistically blends finite-state analysis for shallow syntactic parsing with a machine learning technique. Particularly novel components of this blend are the aggressive analysis, by a complex grammar cascade, targeting considerably more than just partitioning text into chunks; coupled with an extension of the learning component, specially designed to counteract paucity in pre-annotated data with

the ability to train over unannotated data as well as exploit whatever labeled data is available, no matter how small.

We have described our strategy; in retrospect, it is additionally motivated by the encouraging results we report on a series of experiments. Analysis of the results shows that particular adaptations of the strategy are appropriate for the different TimeML components. This is work in progress, and we have noted room for improvement.

References

[AAAI / QA, 2003] AAAI / QA (2003). *New Directions in Question-Answering (Working Papers)*, Stanford, CA. American Association for Artificial Intelligence 2003 Spring Symposium Series.

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

[Ando, 2004b] Ando, R. K. (2004b). Semantic lexicon construction: Learning from unlabeled data via spectral analysis. In *Proceedings of CoNLL-04*.

[Boguraev, 2000] Boguraev, B. (2000). Towards finite-state analysis of lexical cohesion. In *Proceedings of the 3rd International Conference on Finite-State Methods for NLP*, INTEX-3, Liege, Belgium.

[Boguraev and Neff, 2003] Boguraev, B. and Neff, M. (2003). The Talent 5.1 TFst system: User documentation and grammar writing manual. Technical Report RC22976, IBM T.J. Watson Research Center, Yorktown Heights, New York.

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

[Fikes et al., 2003] Fikes, R., Jenkins, J., and Frank, G. (2003). JTP: A system architecture and component library for hybrid reasoning. Technical Report KSL-03-01, Knowledge Systems Laboratory, Stanford University.

[Filatova and Hovy, 2001] Filatova, E. and Hovy, E. (2001). Assigning timestamps to event-clauses. In *Proceedings of the 10th Conference of the European Chapter of the ACL*, Toulouse, France.

[Florian et al., 2004] Florian, R., Hassan, H., Jing, H., Kambhatla, N., Luo, X., Nicolov, N., and Roukos, S. (2004). A statistical model for multilingual entity detection and tracking. In *Proceedings of HLT-NAACL'04*.

[Florian et al., 2003] Florian, R., Ittycheriah, A., Jing, H., and Zhang, T. (2003). Named entity recognition through classifier combination. In *Proceedings of CoNLL-2003*.

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

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

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

[Kennedy and Boguraev, 1996] Kennedy, C. and Boguraev, B. (1996). Anaphora for everyone: Pronominal anaphora resolution without a parser. In *Proceedings of COLING-96 (16th International Conference on Computational Linguistics)*, Copenhagen, DK.

[Levin, 1993] Levin, B. (1993). *English verb classes and alternations*. The University of Chicago Press, Chicago and London.

[Mani et al., 2003] Mani, I., Schiffman, B., and Zhang, J. (2003). Inferring temporal ordering of events in news. In *Human Language Technology/ North American Chapter of the ACL (HLT-NAACL*, Edmonton, Canada.

[Mani and Wilson, 2000] Mani, I. and Wilson, G. (2000). Robust temporal processing of news. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, Hong Kong.

[Neff et al., 2003] Neff, M., Byrd, R., and Boguraev, B. (2003). The Talent system: TEXTRACT architecture and data model. Technical Report RC22973, IBM T.J. Watson Research Center. Originally presented at HLT-NAACL Workshop on Software Engineering and Architectures of Language Technology Systems, Edmonton, Canada.

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

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

[Saurí et al., 2004] Saurí, R., Littman, J., Gaizauskas, R., Setzer, A., and Pustejovsky, J. (2004). TimeML annotation guidelines. Technical report, TERQAS Workshop. Version 1.1, `http://www.cs.brandeis.edu-/~jamesp/arda/time/timeMLdocs/guidetest.pdf`.

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

[TERN, 2004] TERN (2004). The TERN evaluation plan; time expression recognition and normalization. Working papers, TERN 2004 Evaluation Workshop.

[Wood, 1967] Wood, F. T. (1967). *English Prepositional Idioms*. The Macmillan Press Ltd, London and Basingstoke.

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

[Zhang and Johnson, 2003] Zhang, T. and Johnson, D. E. (2003). A robust risk minimization based named entity recognition system. In *Proceedings of CoNLL-2003*, pages 204–207.