Effective Use of TimeBank for TimeML Analysis

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. TimeML is an expressive language for temporal information, but its rich representational properties raise the bar for traditional information extraction methods when applied to the task of text-to-TimeML analysis. We analyse the extent to which TimeBank, the reference corpus for TimeML, supports development of TimeML-compliant analytics. The first release of the corpus exhibits challenging characteristics: small size and some noise. Nonetheless, a particular design of a time annotator trained on TimeBank is able to exploit the data in an implementation which deploys a hybrid analytical strategy of mixing aggressive finite-state processing over linguistic annotations with a state-of-the-art machine learning technique capable of leveraging large amounts of unannotated data. We present our design, in light of encouraging performance results; we also interpret these results in relation to a close analysis of TimeBank's annotation 'profile'. We conclude that even the first release of the corpus is invaluable; we further argue for more infrastructure work needed to create a larger and more robust reference corpus.[1]

Keywords: corpus analysis, TimeBank, TimeML, temporal information extraction, machine learning.

1 Introduction

TimeML was designed [1] to connect the processes of temporal analysis of a text document into a rich, intermediate, representation and its subsequent formalisation by means of an ontology of time [2]. This paper assumes some familiary with TimeML; in essence, the language uses the representational principles of XML markup to annotate the analysis of the core elements in a temporal framework: time expressions, events, and links among these (additionally moderated by temporal connectives, or signals). For details of the markup language for time, readers are referred to [3].

In line with the established methodology of creating community-wide annotated resources, where linguistic analysis is captured by means of a range of tags, and finer-grained specification of analytical detail is expressed by means of suitably defined attributes on these tags, TimeML implements a flexible representational scheme for text markup. At the same time, the language takes

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

F. Schilder et al. (Eds.): Reasoning about Time and Events, LNAI 4795, pp. 41–58, 2007.

the notion of markup to an extreme, developing half-a-dozen entity and relation marking tags—both consuming and non-consuming—and defining a large number of attributes for most of them.

Consequently, the resulting language is both very expressive and very complex. The expressiveness is almost a necessity, arising from the richness of time information and depth of temporal analysis, and addressed from the beginning of the design effort. The complexity is at least in comparison with markup schemes designed for the kinds of "named entities" which have traditionally been at the focus of conventional information extraction (IE) endeavours.

Many markup schemes for IE to date target relatively simple phenomena; unlike TimeML, their design has not been informed by the need to capture the variety and complexity of information required to support inference and reasoning. The extent to which IE can be argued to offer some basis for language understanding can be found in the 'spirit' of the MUC[2] *event scenario* tasks, which instantiate semantic networks [4,5]. However, the mapping of an entire text document to a single template can hardly be regarded as logically complete and coherent, in the sense required and assumed by formal event and/or time ontologies.

More recently, a growing body of work has initiated investigations into the nature of *linguistic annotation*—as a principled description of a linguistic phenomenon of interest (see, for instance, [6]). Such a description, of course, would then be instrumental to a deeper level of analysis and understanding.

Interestingly enough, an early instance of such an annotation effort—with a schema focusing on an identifiable linguistic phenomenon, and not just "named entities" markup—was defined by the ACE[3] event timestamping task, which sought to identify within-sentence event-time links. it could be argued that even within the limited set of event classes defined to be in the scope of ACE, the emphasis in schema design was on the annotation of relational information over a full inventory of temporal relations, and not just that of extents and spans.

In a similar spirit, TimeML aims to capture a much richer set of the temporal characteristics in a text document, so that the intricate temporal linking among all time expressions and events can then get fully mapped onto an ontologically-grounded temporal graph (or its equivalent), cf. [7], [8]. Indeed, such a mapping (see [9] for a sketch) has been one of the guiding principles in the conception and design of TimeML.

The design of TimeML therefore brings both promises—but also challenges—as its representational properties significantly raise the bar for traditional information extraction methods. A particularly relevant question, then, concerns the extent to which TimeML-compliant analysis can be automated: temporal reasoning frameworks crucially require such analysis for any practical understanding of time: "... the [TimeML] annotation scheme itself, due to its closer tie to

[2] Message Understanding Conferences; see `http://www.itl.nist.gov/iad/894.02/related_projects/muc/main.html`

[3] Automatic Content Extraction; see `http://www.nist.gov/speech/tests/ace/index.htm`

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." [8].

Analysis into TimeML is the primary question addressed by this paper. We start from the position stated by Pustejovsky et al. [10] as one of the guiding motivations for developing the TimeBank corpus, which is the primary reference resource for TimeML: it would be regarded as a resource for "training and evaluating algorithms which determine event ordering and time-stamping" (ibid.), as well as providing general-purpose training data for any and all TimeML components. We then demonstrate that small (and somewhat noisy) as it is (compared to guidelines implicitly established by other information extraction tasks relying on annotated data), TimeBank is still the valuable resource that [10] describes.

Our method rests on developing a strategy for time analysis of text specifically informed by the characteristics of TimeBank: a synergistic approach deploying both finite-state (FS) grammars with broad range of analysis and machine learning techniques capable of also leveraging unannotated data. Thus we aim to make maximal use of the information captured by this particular corpus, even if it was not explicitly designed and constructed as a proper training resource.

2 Quantitative and Qualitative Analysis of TimeBank

One of the common characteristics of annotation efforts is that they make, from the outset, infrastructural provisions for the development of a substantial 'reference' corpus, which defines a gold standard ("truth") for the task. The corpus contains materials selected to be representative of the phenomenon of interest; sizes of training and testing samples are carefully considered especially as they depend on the complexity of the task; experienced annotators are used; the corpus is not released until a certain level of inter-annotator agreement is reached. These measures ensure that the reference corpus is of a certain size and quality.

The TimeBank corpus is small. This need not be surprising, given that the TERQAS[4] effort did not commit to producing a 'reference', training-strength, corpus in the sense described above. In fact, TimeBank is almost a 'side effect' of the work: it was largely an exercise in applying the annotation guidelines—as they were being developed—to real texts (news articles, primarily) in order to assess the need for, and then the adequacy of, the language representational devices as they were being designed in the process of TimeML evolution.

Just how small TimeBank is is illustrated by the following statistics. The corpus has only 186 documents, with a total of 68.5K words. As there are no

[4] Temporal and Event Recognition for QA Systems; `http: //www.timeml.org/terqas/index.html`). The TERQAS effort coordinated, over an extended period of time, a series of definitional and follow-up workshops from which emerged the current set of TimeML annotation guidelines.

separate training and test portions, it would need partitioning somehow; if we held out 10% of the corpus as test data, we have barely over 60K words for training.

To put this into perspective, this is order of magnitude less than other standard training corpora in the NLP community: the Penn Treebank corpus[5] for part-of-speech tagging (arguably a simpler task than TimeML component analysis) contains more than 1M words—which makes it over 16 times larger than TimeBank; the CoNLL'03 named entity chunking task[6] is defined by means of a training set with over 200K words. A task closely related to time analysis is ACE's TERN (Temporal Expression Recognition and Normalisation)[7]. TERN only focuses on TIMEX2 (TIMEX3, which extends the TIMEX2 tag [3], is just one of half-a-dozen TimeML components); even so, the TERN training set is almost 800 documents/300K words-strong.

Boguraev et al. [11] offer a detailed quantitative and qualitative analysis of the TimeBank corpus, in its original version—which was the basis for the experiments and results reported in this paper. In general, the observation is that the combination of the small size of TimeBank, the uneven distribution of TimeML components, and the erroneous annotation introduced by mixture of infrastructure issues and annotation methodology, lead to some significant challenges in using the corpus as a training resource.

Consider, for instance, the extreme paucity of positive examples over a range of categories. Fig. 1 (reproduced here, for convenience of reference, from [11]) shows the distribution of TLINK and EVENT types. These are the 'targets' of relational time analysis, capturing the temporal semantics above time expressions. As such, they are crucial for any analytical device.

The numbers in the figure illustrate the highly uneven distribution of this category data. The numbers also reveal some of the variety and complexity of TimeML annotation: the extensive typing of EVENTs, TIMEX3's and LINKs introduces even more classes in an operational TimeML typology. Thus an event recognition and typing task is, in effect, concerned with partitioning recognised events into 7 categories (as we shall see in Section 5.2, a particular implementation of such a partitioning is realised as $(2k + 1)$-way classification task, where $k = 7$ in our case). Similarly, for TLINK analysis the relevant comparison is to consider that in contrast to, for instance, the CoNLL'03 named entity recognition task—with training data containing 23K examples of named entities belonging to just 4 categories, TimeBank offers less than 2K examples of TLINKs, which, however, range over 13 category types.

The analysis in [11] additionally discusses the sources of noise in the first release of TimeBank. Broadly speaking, there are three different categories of error: errors due to failures in the annotation infrastructure, errors resulting from broad interpretation of the annotation guidelines, and errors due to the inherent complexity of the annotation task (further compounded by underspecification in

[5] See `http://www.cis.upenn.edu/treebank/home.html`
[6] See `http://cnts.uia.ac.be/conll2003/ner/`
[7] See `http://timex2.mitre.org/tern.html`

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

Fig. 1. Distribution of (some) TimeML component types. Note that the count of 1451 TLINKs refers only to the TLINKs between an event and a temporal expression, itself in the body of a document. (TLINKs with TIMEX3's in metadata are not counted here.)

the guidelines). The reader is referred to that discussion, because it is important, for correctly situating our experiments and interpretating the results, to have an appreciation of the degree of noise which is at a level above what typically might be expected of a training resource.

Parenthetically, we observe that the kind of detailed analysis presented in [11]—itself motivated by the desire to understand how to interpret the performance figures reported in this paper—was itself the basis for a focused effort to revise and clean up the TimeBank corpus, which is currently distributed (as Version 1.2) through the offices of the Linguistic Data Consortium.

3 Challenges for TimeML Analysis

It is clear that temporal annotation is a very complex problem: TimeML was developed precisely to address the issues of complexity and to provide a representational framework capable of capturing the richness of analysis required. One consequence of this is the pervasiveness of relational data which is integral to the underlying representation: all links are, notationally, relations connecting events with other events or temporal expressions. As recent work in relation finding information extraction shows (in particular, in the context of the ACE program), the task requires both some linguistic analysis of text and the definition of complex learning models, typically going beyond just token sequences.

Additionally, as the previous section shows, a different degree of complexity is introduced by the size (and coverage) characteristics of TimeBank. While it may be reasonable to take a position that in our investigation we will focus on those TimeML components which are relatively more prevalent in the data (*e.g.* TLINKs over ALINKs and SLINKs), we still need to address the problem of insufficient training data. Our position thus is that in addition to deploying sophisticated feature generators, we crucially need to leverage machine learning technology capable of exploiting unlabeled data.

Our strategy for TimeML analysis of text develops a hybrid approach utilising both finite-state (FS) grammars over linguistic annotations and machine learning (ML) techniques incorporating a novel learning strategy from large volumes of unlabeled data. The respective strengths of these technologies are well suited for the challenges of the task: complexity of analysis, need for some syntactic and discourse processing, and relative paucity of examples of TimeML-style annotation.

The initial targets of our analysis are TIMEX3 (with attributes), EVENT (plus type), and TLINK (plus type, and limited to links between events and time expressions); see Section 2 and Fig. 1. This kind of limitation is imposed largely by the distributional properties of TimeML components annotated in TimeBank (as discussed in Section 2 earlier); but it is also motivated by the observation that to be practically useful to a reasoner, a time analysis framework would need to support, minimally, time stamping and temporal ordering of events. As this is work in progress, the description below offers more details specifically on identifying TIMEX3 expressions, marking and typing EVENTs, and associating (some of these) with TIMEX3 tags (typing the links, as appropriate).

All of these subtasks have components which can be naturally aligned with one or the other of our strategic toolkits. Thus TIMEX3 expressions are intrinsically amenable to FS description, and a grammar-based approach is well-suited to interfacing to the task of TIMEX3 normalisation (*i.e.* instantiating its value). On the other hand, certain attributes of a TIMEX3 (such as temporalFunction, valueFromFunction, functionInDocument) can be assigned by a machine learning component. 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 Section 4). To complement such analysis, an ML approach can, using suitable classification methods, cast the problem of marking (and typing) EVENTs as chunking (Section 5.2). As we will see later, a TLINK classifier crucially relies on features derived from the configurational characteristics of a syntactic parse; a result in line with recent work which shows that mid-to-high-level syntactic parsing—typically derived by FS cascades—can produce rich features for classifiers.

In summary, we address the challenges of the TimeBank corpus by combining FS grammars for temporal expressions, embedded in a shallow parser adapted for time analysis, with machine learning trained with TimeBank and unannotated corpora.

4 Finite State Devices for Temporal Analysis

Temporal expressions conform to a set of regular patterns, amenable to grammar-based description. 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 as unit, point, period, relation, *etc*; typically, these will be further decomposed and typed into *e.g.* month, day, year (for a unit); or interval or duration (for a period).

Temporal expressions are characterised by "local" properties—granularity (*e.g.* month, day, *etc*), cardinality, ref_direction(*e.g.* prior, or subsequent to "now"), and so forth—which are intrinsic to their temporal nature, but not directly related to TIMEX3 attributes. Fine-grained analysis of temporal expressions, instantiating such local propertiesis is crucially required for normalising a TIMEX3: consider, for example, that representing *e.g. "the last five years"* as illustrated in Fig. 2 below greatly facilitates the derivation of a value (in this case "5PY") for the TIMEX3 value attribute.

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

Fig. 2. Analysis of a time expression in terms of local attributes

In effect, 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*).

It is important to separate the processes of recognition of the span of a TIMEX3 expression from local attribute instantiation for that expression. There is nothing intrinsic to the recognition which necessitates a grammar-based description in preference to a statistical model (as the TERN evaluation exercise demonstrated [12]). However, local attributes (as exemplified above) are necessary for the interpretation rules deriving TIMEX3 value.

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').

In addition to *parsing of temporal expressions*, FS devices are deployed for *shallow parsing for feature generation*. We build upon prior work [13], which

showed how substantial discourse processing can be carried out from a shallow syntactic base, and derived by means of FS cascading.

Our grammars interleave syntactic analysis with named entity extraction. In particular, they define temporal expressions—as well as other TimeML components, namely events and signals—in terms of *linguistic* units, as opposed to simply lexical cues (as many temporal taggers to date do). The focus on linguistic description 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. [14]), 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 the configurational properties of a prepositional phrase—that the nominal argument (*"war"*) of the temporal preposition (*"during"*) is (most likely) 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, from whose point of view an EVENT annotation is just another feature. Indeed, the only use of syntactic analysis beyond the TIMEX3 parser is to populate a feature space 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). The scheme we use (Section 5) requires additionally some semantic typing, knowledge of boundaries of longer syntactic units (typically a variety of clauses), and some grammatical function. Fig. 3 illustrates the nature of the FS cascade output.

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 (a label like GrmEventOccurrence denotes a hypothesis, generated by the syntactic grammars, that *"earned"* is an occurrence type EVENT). 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.

```
[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]
```

Fig. 3. Shallow syntactic analysis (simplified) from finite-state parsing

TimeML parsing is thus a bifurcated process of TimeML components recognition: TIMEX3's are marked by FS grammars; SIGNALs, EVENTs and LINKs are putatively marked by the grammars, but the final authority on their identification are classification models built from analysis of both TimeBank and large unannotated corpora. *Features* for these models are derived, as we shall see below, from common strategies for exploiting local context, as well as from mining the results—both mark-up and configurational—of the FS grammar cascading.

5 Classification Models for Temporal Analysis

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.

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. This is particularly important in an investigation like ours, where we do not know *a priori* what kinds of features and feature sets would turn out to be most productive.

For our experiments (Section 6), we use the *Robust Risk Minimization* (RRM) classifier [15], a linear classifier, which has independently been shown useful for a number of text analysis tasks such as syntactic chunking [15], named entity chunking [16,17,18], and part-of-speech tagging [19].

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 that the algorithms change. We show how choice of features affects performance in Section 6.

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*). Before we describe (later in this section) the essence of our feature design for EVENT and TLINK recognition,[8] we briefly outline word profiling as the enabling technique for counteracting the paucity of training data in TimeBank.

5.1 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 (Section 2). This

[8] 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, independent of whether it is tagged as SIGNAL or not.

characteristic of size is potentially a limiting factor in supervised 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 [19]. Word profiling identifies, extracts, and manipulates information that characterizes words from unannotated corpora; it does this, in essence, by collecting and compressing feature frequencies from the corpus, a process which maps the commonly used feature vectors to frequency-encoded context vectors.

More precisely, word profiling turns co-occurrence counts of words and features (within certain syntactic configurations: *e.g.* 'next word', 'within a phrase', 'head of subject', *etc*) into new feature vectors. Note that this requires preanalysis of the unannotated corpus. For example, observing—in that corpus—that nouns like *"extinction"* and *"explosion"* are often used as syntactic subject to *"occur"*, and that *"happen"*'s subjects contain *"earthquake"* and *"explosion"*, helps to predict that *"explosion"*, *"extinction"*, and *"earthquake"* all function like event nominals. Such a prediction is motivated by the parallel observation about the preponderance, in the annotated corpus, of event nominals in subject position to *"occur"* and *"happen"* . In Section 6.2, we demonstrate the effectiveness of word profiling, specifically for EVENT recognition.

5.2 EVENT Recognition as a Classification Problem

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.

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

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 encoding 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 [19]. The features we extract are:

- token, capitalisation, 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);

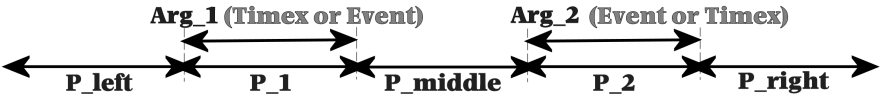

Fig. 4. Partitions for TLINK classifier segmentation

- token tags in 2-token window to the left;
- tri-grams of POS, capitalisation, and word ending;
- tri-grams of POS, capitalisation, and left tag.

5.3 TLINK Recognition as a Classification Problem

TLINK is a relation between events and time expressions which can link two EVENTs, two TIMEX3's, or an EVENT and a TIMEX3. As we stipulated earlier (Section 3), presently we focus on TLINKs between events and time expressions.

As a relational link, TLINK does not naturally fit the tagging abstraction underlying the chunking problem, as 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*). The adjustment term '+1' is for the *negative* class, which indicates that the pair does not have any kind of temporal link relation.

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 (Fig. 4).

We also consider *segments* (*i.e.* 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 in Fig. 3 (p. 48), `svoClause` is the smallest segment containing both an EVENT and a TIMEX3, which is indicative of (or at least does not prohibit) a direct syntactic relation between the two. In the next example (Fig. 5), the TIMEX3 and EVENT chunks are contained in different clauses (a

```
[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]
```

Fig. 5. Syntactic configuration discouraging of a TLINK

`thatClause` and a `svoClause`, respectively), which structurally prohibits a TLINK relation between the two. 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

In line with our current investigation focus (as defined in Section 3), we present here performance results on recognition and typing of TIMEX3, EVENT and TLINK only. Our primary objective here 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. This is the first attempt to build a TimeML-compliant analyser which addresses a more or less full complement of TimeML components; thus 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.

6.1 TIMEX Recognition and Typing

Fig. 6 presents performance results of our TIMEX3 analysis subsystem. Experiments were carried out under different settings. "Span" refers to strict match of both boundaries (the extent) of a TIMEX3 expression; "sloppy" admits time expressions recognised by the FS grammars as long as their right boundary is the same as the reference expression in TimeBank. (One of the observations from the quality analysis of TimeBank reported in [11] is that the corpus is inconsistent with respect to whether some 'left boundary' items—determiners, pre-determiners, and so forth—are marked as a part of the time expression or not; the "sloppy" setting tries to account for this somewhat). As of the time or writing, there are no published results for full TimeML-compliant analysis. We offer here only indirect assessment of our TIMEX3 analysis task, by observing that the numbers for extent marking are not very far from the best systems performance reported at the TERN conference. Of course, given the different definitions of TIMEX2 and TIMEX3, as well as TimeBank's relatively 'cavalier' attitude with respect to TIMEX3's left boundary, the comparison is not very

Task	P	R	F
Span	77.6	86.1	81.7
Span ('sloppy')	85.2	95.2	89.6
			Accuracy
Type (given 'true' span)			81.5
Span + type	64.5	71.6	67.9
Span ('sloppy') + type	70.1	77.8	73.7

Fig. 6. TIMEX3 analysis results, with/without typing. Typing carried out after/ simultaneously with span marking.

meaningful; still, it is indicative of some level of grammar coverage, especially given the incommensurate sizes of the TERN training data and the TimeBank corpus (Section 2).

While TIMEX3 spans are determined by grammars, we use a classifier to type the time expressions. Again, this decision was motivated largely by observing some inconsistencies in type assignment in the corpus, and we felt that, for the purposes of strictly matching the data, machine learning was a more fitting approach to try first (we are yet to compare the typing results presented here with typing by the FS grammars; such a comparison is tied somewhat to getting a better understanding of the quality of annotations in TimeBank). The TIMEX3 typing classifier (second segment of Fig. 6) is configured to use "true" TIMEX3 spans, as per TimeBank, as data points, to which it assigns a category (type) label; thus the table gives a single accuracy measure.

Finally, we report on a joint task, which combines (in sequence) extent marking by FS grammars and type determination as classification process over given spans (this classification task, and features, are defined similarly to the IEO scheme used for EVENT extraction and typing, in without-word-profiling setting; see Section 5.2). In effect, the results here confirm the intuition that imperfect subtasks individually contribute to cumulative degradation of performance.

6.2 EVENT Recognition and Typing

The example analysis in Fig. 3, and the description of features used for the EVENT classification task (Section 5.2) demonstrates how local information and syntactic environment both contribute to the feature generation process. Fig. 7 shows performance results with and without word profiling for exploiting an unannotated corpus.

For the word profiling experiments, we extracted feature co-occurrence counts from 40 million words of 1991 *Wall Street Journal* articles. The proposed event chunks are counted as correct only when both the chunk boundaries and event

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

Fig. 7. EVENT extraction results, with/without typing. Numbers in parentheses show contribution of word profiling, over using basic features only.

types are correct. 64.0% F-measure is lower than typical performance of, for instance, named entity chunking; this result is indicative of the effects of insufficient training data. On the other hand, a strongly positive indicator here is the fact that word profiling clearly improves performance. In a different setting, when we train the EVENT classifiers without typing, we obtain 80.3% F-measure. This confirms the intuition that the EVENT typing task is inherently complex, and requires more training data.

6.3 TLINK Recognition and Typing

In this experimental setting, we only consider the pairings of EVENT and TIMEX3 which appear within a certain distance in the same sentences (as we will see shortly, this hardly reduces the problem space).[9]

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 Fig. 8. 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% (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 against 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. When we focus on the pairings within a 4-tokens window, we achieve 81.8% F-measure without typing of TLINKs, and 58.8% with typing. (The task

[9] To evaluate the TLINK classifier alone, we use the EVENT and TIMEX3 annotations in TimeBank. Also, note that the focus on links within a sentence span naturally excludes TLINKs with time expressions in document metadata.

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)

Fig. 8. TLINK extraction results, with/without typing. Parentheses show positive contribution of grammar-derived features, over using basic features only. Baseline method posts TLINKs over 'close' pairs of EVENTs and TIMEX3's.

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, for instance,, 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 major cause of the relatively low performance of TLINK typing is the difficulty in distinguishing between during and is_included link types.

7 Conclusion

We have used the task of TimeML-compliant parsing to experiment with a specially developed strategy for leveraging minuscule amounts of training data. The strategy synergistically blends finite-state analysis for shallow syntactic parsing with a machine learning technique. The potential for such synergistic approaches to complex analytical problems is clear, especially in situations where reference data—in sufficient quantity, and/or quality—is hard to come by.

This paper highlights two aspects of this blend. We carry out aggressive analysis, by a complex grammar cascade, aiming at considerably more than just partitioning text into chunks: the analysis targets both intrinsic characteristics of temporal expressions, as well as higher-order syntactic configurations used to derive features for a machine learning component. The learning component itself is enhanced by a mechanism specifically 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.

The extreme paucity of the available reference data correlates with the performance results, in particular where the novel components of EVENT and TLINK analysis are targeted, as they appear to fall short of expectations in line with current state-of-the-art information extraction capabilities. Our results are further explained by the inherently noisy nature characteristic of the TimeBank corpus. However, given that the corpus was not designed and populated using rigorous methods for generating training data, our experience is indicative of the effectiveness of a hybrid analytical approach.

Direct comparison of the results reported here with related work is not yet possible. Ours is the first systematic attempt at TimeML-compliant analysis, aiming at a more or less full complement of TimeML components: thus there are no comparable results in the literature.

Mani et al. [20] discuss some pioneering work in linking events with times, and ordering events, suggestive 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. Most recently, the TARSQI project has been developing strategies and heuristics for particular subsets of TimeML components [21]; again, there is no basis for direct comparison, as only partial overlap exists between the phenomena and attributes targeted by that work and ours (but see [11] for some in-depth analysis of complementary analytic strategies). For this reason, as well as because TARSQI does not explicitly focus on investigating the utility of TimeBank as a training resource, it is not constructive to attempt comparative assessment.

One thing our work makes especially clear is that, given the ability to use unannotated corpora in conjunction with TimeBank to develop a more accurate and felicitous TimeML models, even small improvements to the corpus would significantly boost performance. The corpus would benefit substantially from the application of rigorous methodology for compiling training data. Even a relatively minor effort of cleaning up the existing data would improve performance: this is confirmed by considering the results presented in Section 6 and the corpus characteristics highlighted in Section 2.

A cleanup operation—largely focused on fixing both the errors of omission and commission in the original TimeBank—has now been carried out: TimeBank Version 1.2 represents a considerable improvement over TimeBank 1.1, with respect to largely removing the noise in the first release [11]. TimeBank 1.2 is available through the offices of Linguistic Data Consortium. Future work, of further use to the community, would be an effort to create a larger TimeBank which—by virtue of the systematic methods of developing an annotated corpus within an established set of annotation guidelines—will truly become the widely usable reference resource envisaged from the outset of the TimeML definition and by more recent standardisation efforts [22].

References

1. Pustejovsky, J., Castaño, J., Ingria, R., Saurí, R., Gaizauskas, R., Setzer, A., Katz, G., Radev, D.: TimeML: Robust specification of event and temporal expressions in text. In: AAAI Spring Symposium on New Directions in Question-Answering (Working Papers), Stanford, CA, pp. 28–34 (2003)
2. Hobbs, J., Pan, F.: An ontology of time for the semantic web. TALIP Special Issue on Spatial and Temporal Information Processing 3(1), 66–85 (2004)
3. Saurí, R., Littman, J., Knippen, B., Gaizauskas, R., Setzer, A., Pustejovsky, J.: TimeML annotation guidelines. Technical report, TERQAS Workshop (2005), Version 1.4, (date of citation: February 02, 2006) `http://timeml.org/site/publications/timeMLdocs/AnnGuide_1.2.1.pdf`
4. Advanced Research Projects Agency: In: Proceedings of the Sixth Message Understanding Conference (muc-6), Advanced Research Projects Agency, Software and Intelligent Systems Technology Office (1995)
5. Advanced Research Projects Agency: In: Proceedings of the Seventh Message Understanding Conference (muc-7), Advanced Research Projects Agency, Software and Intelligent Systems Technology Office (1998)
6. Boguraev, B., Ide, N., Meyers, A., Nariyama, S., Stede, M., Wiebe, J., Wilcock, G.: Linguistic Annotation Workshop (the LAW); ACL-2007, Prague, The Czech Republic, Association for Computational Linguistics (June 2007)
7. Fikes, R., Jenkins, J., Frank, G.: JTP: A system architecture and component library for hybrid reasoning. Technical Report KSL-03-01, Knowledge Systems Laboratory, Stanford University (2003)
8. Han, B., Lavie, A.: A framework for resolution of time in natural language. TALIP Special Issue on Spatial and Temporal Information Processing 3(1), 11–35 (2004)
9. Hobbs, J., Pustejovsky, J.: Annotating and reasoning about time and events. In: AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning, Stanford, CA (March 2004)
10. Pustejovsky, J., Hanks, P., Saurí, R., See, A., Gaizauskas, R., Setzer, A., Radev, D., Sundheim, B., Day, D., Ferro, L., Lazo, M.: The Timebank corpus. In: McEnery, T. (ed.) Corpus Linguistics, Lancaster, pp. 647–656 (2003)
11. Boguraev, B., Pustejovsky, J., Ando, R., Verhagen, M.: Evolution of TimeBank as a community resource for TimeML parsing. Language Resources and Evaluation (Forthcoming 2007)
12. DARPA TIDES (Translingual Information Detection, Extraction and Summarization): The TERN evaluation plan; time expression recognition and normalization. In: Working papers, TERN Evaluation Workshop (2004), (date of citation: July 12, 2005) `http://timex2.mitre.org/tern.html`
13. Kennedy, C., Boguraev, B.: Anaphora for everyone: Pronominal anaphora resolution without a parser. In: Proceedings of COLING 1996. 16th International Conference on Computational Linguistics, Copenhagen, DK (1996)
14. Schilder, F., Habel, C.: Temporal information extraction for temporal QA. In: AAAI Spring Symposium on New Directions in Question-Answering (Working Papers), Stanford, CA, pp. 35–44 (2003)
15. Zhang, T., Damerau, F., Johnson, D.E.: Text chunking based on a generalization of Winnow. Journal of Machine Learning Research 2, 615–637 (2002)
16. Florian, R., Ittycheriah, A., Jing, H., Zhang, T.: Named entity recognition through classifier combination. In: Proceedings of CoNLL-2003 (2003)

17. Zhang, T., Johnson, D.E.: A robust risk minimization based named entity recognition system. In: Proceedings of CoNLL-2003, pp. 204–207 (2003)
18. Florian, R., Hassan, H., Jing, H., Kambhatla, N., Luo, X., Nicolov, N., Roukos, S.: A statistical model for multilingual entity detection and tracking. In: Proceedings of HLT-NAACL 2004 (2004)
19. Ando, R.K.: Exploiting unannotated corpora for tagging and chunking. Proceedings of ACL 2004 (2004)
20. Mani, I., Pustejovsky, J., Sundheim, B.: Introduction: special issue on temporal information processing. ACM Transactions Asian Language Information Processing 3(1), 1–10 (2004)
21. Verhagen, M., Mani, I., Sauri, R., Littman, J., Knippen, R., Jang, S.B., Rumshisky, A., Phillips, J., Pustejovsky, J.: Automating temporal annotation with tarsqi. In: ACL 2005. 43rd Annual Meeting of the Association for Computational Linguistics, Ann Arbor, Michigan, (Poster/Demo) (2005)
22. Lee, K., Pustejovsky, J., Boguraev, B.: Towards an international standard for annotating temporal information. In: Third International Conference on Terminology, Standardization and Technology Transfer, Beijing, China, ISO TC/37 and SC (August (2006)