A Cascaded Machine Learning Approach to Interpreting Temporal Expressions

David Ahn Joris van Rantwijk Maarten de Rijke
ISLA, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
{ahn, rantwijk, mdr}@science.uva.nl

Abstract

A new architecture for identifying and interpreting temporal expressions is introduced, in which the large set of complex hand-crafted rules standard in systems for this task is replaced by a series of machine learned classifiers and a much smaller set of context-independent semantic composition rules. Experiments with the TERN 2004 data set demonstrate that overall system performance is comparable to the state-of-the-art, and that normalization performance is particularly good.

1 Introduction

In order to fully understand a piece of text, we must understand its temporal structure. The first step toward such an understanding is identifying explicit references to time. We focus on the task of automatically annotating temporal expressions (or *timexes*)—both identifying them in text and interpreting them to determine what times they refer to. Timex annotation is more than normalizing date expressions. First, time consists of more than calendar dates and clock times—it also includes points of finer and coarser granularity, durations, and sets of times. Second, the expressions that refer to time are not just full date and time expressions—they may be underspecified, ambiguous, and anaphoric.

Building a system for the full timex identification and interpretation task can be tedious, requiring a great deal of manual effort. The 2004 Temporal Expression Recognition and Normalization (TERN) evaluation[1] evaluated systems on two tasks: timex *recognition* (identification) alone and recognition and *normalization* (interpretation) together. All the full-task systems were rule-based systems; the top performing full-task system uses in excess of one thousand hand-crafted rules, which probe words and their contexts in order to both identify timexes and to assemble information necessary to interpret them (Negri and Marseglia, 2004). By contrast, machine learned systems dominated the recognition-only task and even achieved slightly better recognition scores than their rule-based counterparts.

We seek to demonstrate that a timex annotation system that performs both recognition and normalization need not be a tangle of rules that serve double duty for identification and interpretation and that mix up context-dependent and context-independent processing. We propose a novel architecture that clearly separates syntactic, semantic, and pragmatic processing and factors out context-dependent from context-independent processing. Factoring out context-dependent disambiguation into separate classification tasks introduces the opportunity for using machine learning, which supports our main goal: building a portable, trainable timex annotation system in which the role of hand-crafted rules is minimized. The system we present here (available from http://ilps.science.uva.nl/Resources/timextag/) achieves the goal of making use of only a small set of hand-crafted, context-independent rules to achieve state-of-the-art normalization performance.

In the following section, we define what a timex is. We give an overview of our system architecture in §3 and describe the components in §4–7. §8 provides an evaluation of our system on the full timex annotation task, and we conclude in §9.

[1] http://timex2.mitre.org/tern.html

2 What is a timex?

Temporal semantics receives a great deal of attention in the semantics literature (cf. (Mani et al., 2005)), but the focus is generally on verbal semantics (i.e., tense and aspect). In determining what a timex is and how one should be normalized, we simply follow the TIDES TIMEX2 standard for timex annotation (Ferro et al., 2004). According to this standard, timexes are phrases or words that refer to times, where times may be points or durations, or sets of points or durations. Points are more than just instanteous moments in time—a point may also be a time with some duration, as long as it spans a single unit of some temporal granularity. Whether a timex refers to a point or a duration is a question of perspective rather than of ontology. A point-referring timex such as *October 18, 2006* refers to an interval of one day as an atom at the granularity of a day. A duration-referring timex such as *the whole day* may refer to the same temporal interval, but it focuses on the durative nature of this interval.

In addition to specifying which phrases are timexes, the TIMEX2 standard also provides a set of attributes for normalizing these timexes. We focus on the VAL attribute, which takes values that are an extension of the ISO-8601 standard for representing time (ISO, 1997). TIMEX2 VAL attributes can take one of three basic types of values:

Points are expressed as a string matching the pattern `dddd-dd-ddTdd:dd:dd.d+`, where `d` indicates a digit. Such a string is to be interpreted as *year*-*month*-*date*T*hour*:*minute*:*seconds*, and may be truncated from the right, indicating points of coarser granularity. Any place may be filled with a placeholder `X`, which indicates an unknown or vague value, and there are also a handful of token values (character strings) for seasons and parts of the day which may substitute for months and times. There is also an alternate week-based format `dddd-Wdd-d`, interpreted as *year*-W*week number*-*day of the week*.

Durations are expressed as a string matching the pattern `Pd+u` or `PTd+u`, where `d+` indicates one or more digits and `u` indicates a unit token (such as `Y` for years). A placeholder `X` may be used instead of a number to indicate vagueness.

Vague points: `past_ref`, `present_ref`, `future_ref`.

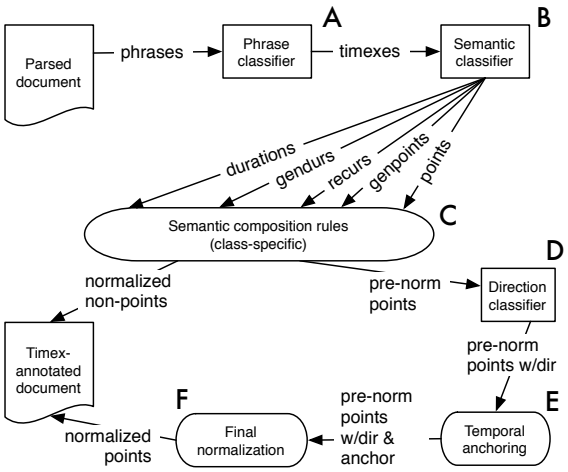

Figure 1: Timex annotation architecture (letters for ease of reference).

The other attribute which we address in this paper is the boolean-valued SET attribute; a SET timex is one that refers to a recurring time. The remaining attributes are MOD, ANCHOR_VAL, and ANCHOR_DIR; our system produces values for these attributes, but we do not address them in this paper.

The TIMEX2 annotation standard has been used to create several manually annotated corpora. For the experiments we present in this paper, we use the corpora annotated for the TERN 2004 evaluation (Ferro, 2004). These consist of a training set of 511 documents of newswire and broadcast news transcripts, with 5326 TIMEX2s, and a test set of 192 similar documents, with 1828 TIMEX2s.

3 Architecture

The architecture of our timex annotation system is depicted in Fig. 1. Our system begins with parsed documents as input. Our recognition module is a machine learned classifier (A); it is described in §4.

Phrases that have been classified as timexes are then sent to the semantic class classifier (B). Semantic class disambiguation is the first point at which context dependence enters into timex interpretation. While some timexes are unambiguous with respect to whether they refer to a point, a duration, or a set, many timexes are semantically ambiguous and can only be disambiguated in context. The machine learned classifier for this task is described in §5.

Based on the class assigned by the semantic class

classifier, the semantic composition component (C) generates (underspecified) semantic representations using class-specific, context-independent rules. The rules we use are simple pattern-matching rules that map lexical items or sequences of lexical items within a timex to semantic representations. We describe the semantic composition component in §6.

For most classes of timexes, the semantic composition component generates a semantic representation that can be directly translated into a normalized value. Timexes that refer to specific points are the only exception. While some point timexes are fully qualified, and thus also directly normalizable, many need to be anchored to another time in context in order to be fully normalized. Thus, context dependence again enters the timex interpretation process, and now in two ways. One is obvious: these referential timexes, which need a temporal anchor, have to find it in context. This task requires a reference resolution process (E), which is described in §7.1.

The second ambiguity regards the relation between a referential timex and its anchor. Referential timexes, like anaphoric definites, relate to their anchors through a bridging relation, which is determined primarily by the content of the timex—e.g., *two years later* refers to a point two years after its anchor. For some referential timexes, though, the direction of the relation (before or after the anchor) is not specified. The machine learned classifier (D) resolves this ambiguity; see §7.2.

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

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

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

4 Component A: Recognizing timexes

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

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

We treat timex recognition as a binary *phrase* classification task: syntactic constituents are classified as timexes or non-timexes. We restrict classification to the following phrase types and lexical categories (based on (Ferro et al., 2004, §5)): NP, ADVP, ADJP, NN, NNP, JJ, CD, RB, and PP.[2] In order to identify candidate phrases and to extract

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

	Identification			Exact match		
	prec	rec	F	prec	rec	F
TEXT	0.912	0.786	0.844	0.850	0.732	0.787
DOC	0.929	0.813	0.867	0.878	0.769	0.819
BRO	0.973	0.891	0.930	0.905	0.829	0.865
BFT	0.976	0.880	0.926	0.885	0.798	0.839

Table 1: Recognition results: Identification.

parse-based features, we parse the TEXT elements of our documents with the Charniak parser (Charniak, 2000). Because of both parser and annotator errors, only 90.2% of the timexes in the training data align exactly with a parse, which gives an estimated upper-bound on recall using this method.

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

As with all our experiments in this paper, we train on the TERN training corpus and test on the test corpus. Our scores (precision, recall and F-measure for both identification (i.e., overlap) and exact-match) are given in Table 1, along with the scores of the best recognition-only (BRO) and full-task (BFT) TERN 2004 systems. Since our phrase classification method is only applied within document TEXT elements, we also present results using both our RE-based document metadata tagger and our phrase classifier for full documents (DOC). Only these scores can be compared with the TERN scores.

Our scores using this method approach those of the best systems, but there is still a gap, which, as we see in §8, affects our overall task performance.

5 Component B: Semantic classification

Timexes may refer to points, durations, or recurrences. While some timexes refer unambiguously to one of these, many timexes are ambiguous between two or even three of these (see (Hitzeman, 1993) for a theoretical semantic perspective on this ambiguity). Timexes may also refer generically or vaguely, which is another source of ambiguity.

While the TIMEX2 standard does not explicitly specify semantic classes in its annotations, the semantic classes we distinguish for our normalization system can be easily inferred from the form of the values of the attributes that are annotated, as follows:
Recurrence (recur): SET attribute set to true
Generic or vague duration (gendur): VAL begins with PX or PTX
Duration: VAL begins with P[0-9] or PT[0-9]
Generic or vague point (genpoint): Three possibilities: time-of-day w/o associated date expression (VAL begins with T[0-9]); general reference to past, present, or future (VAL is one of the vague tokens); date expression with unspecified high-order position (i.e., millennium position is X)
Point: Date expression with specified high-order position (may be precise or not—i.e., may include X at other positions—also may be of any granularity, from millennium down to hundredths of a second).

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

We again use the LIBSVM linear kernel for classification, with the same features as for recognition. Even though some timexes are unambiguous with respect to semantic class, we train the classifier over all timexes, in the expectation that the contexts of unambiguous timexes will be similar enough to those of ambiguous timexes of the same class to help in classification. We compare the performance of our machine learned classifier to a heuristic baseline classifier that uses the head of the timex and the presence of numbers, names, and certain modifiers within the timex to decide how to classify it.

Table 2 gives the error rates, per class and overall, for the baseline and learned classifiers over phrase-aligned gold-standard timexes. The machine learned classifier halves the error rate of the baseline, mostly as a result of better performance on the duration and point classes. In §8, we see how this improvement in classification affects end-to-end performance.

Mani and Wilson (2000) and Ahn et al. (2005b)

classifier	overall	null	duration	...
BL	0.2085	1.0000	0.2534	...
SVM	0.1078	0.4143	0.1507	...
class dist	1290	70	146	...
...	gendur	genpoint	point	recur
...	0.0204	0.1462	0.1322	0.6087
...	0.1020	0.1462	0.0496	0.2174
...	49	253	726	46

Table 2: Error rates: semantic class.

class	rules	example
dur	13	`Numeric -? (UNIT | UNITS)`
gendur	3	`(UNIT | UNITS)`
genpt	21	`(NUM24 | NUMWORD) o ' clock`
point	31	`^ Approx? DAYNAME? MONTHNAME .? Num31OrRank ,? YearNum`
recur	11	`(every | per) Numeric UNITS`
misc	10	`NUMWORD ((and | -)? NUMWORD)*`

Table 3: Distribution of semantic composition rules.

also perform limited semantic class disambiguation. Both use machine learned classifiers to distinguish specific and generic uses of *today*, and Ahn et al. also use a machine learned classifier to disambiguate timexes between a point and a duration reading. Their error rate for this task is 27%, but since a set of heuristics is first used to select just ambiguous timexes, this score cannot be compared to ours.

6 Component C: Semantic composition

The semantic composition module uses context-independent, class-specific rules to compute for each timex an underspecified representation—a typed feature structure that depends on the timex's semantic class (features include unit and value for durations, year, month, date, and referential class for points; cf. (Dale and Mazur, 2006)). As the rules are not responsible for identification or class or direction disambiguation, they are fewer in number and simpler than in other systems (cf. 1000+ in (Negri and Marseglia, 2004)). Each rule consists of an RE-pattern, which may refer to a small lexicon of names, units, and numeric words, and is applied using a custom transducer. In total, there are 89 rules; Table 3 gives the distribution of rules and an example rule for each class. Tokens in `ALLCAPS` indicate lexical classes; tokens in `MixedCase` indicate other rules; and tokens in `lowercase` indicate lexical items.

7 Temporal anchors

Some point timexes are fully qualified, while others require a reference time, or temporal anchor, to be fully normalized.[3] There are three ways in which a temporal anchor is chosen for a timex. Some timexes, such as *today*, *three years ago*, and *next week*, are deictic and anchored to the time of speech (for us, the document timestamp). Others, such as *two months earlier* and *the next week*, are anaphoric and anchored to a salient time in discourse, just like an anaphoric pronoun or definite. The distinction between deictic and anaphoric timexes is not always clear-cut, since many anaphoric timexes, in the absence of an appropriate antecedent, are anchored deictically. A timex may also contain its own anchor: e.g., *two days after May 3*, whose anchor is the embedded anaphoric timex *May 3*.

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

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

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

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

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

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

7.1 Component E: Temporal anchor tracking

Since temporal anchors are not annotated in the TIMEX2 standard, our system uses a simple heuristic method for temporal anchoring (cf. (Wiebe et al., 1997), who use a more complex rule-based system for timex anchoring in scheduling dialogues). Since we distinguish deictic and anaphoric timexes during semantic composition, we use a combination of two methods: for deictic timexes, the document timestamp is used, and for (some) anaphoric timexes, the most recent point timex, if it is fine-grained enough, is used as the temporal anchor (otherwise, the document timestamp is used). Because the documents in our corpora are short news texts, we actually treat anaphoric name-like points as deictic and use the most recent timex only for anaphoric offsets.

7.2 Component D: Direction classification

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

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

We again use the LIBSVM linear kernel for classification. We add two sets of features to those used for recognition and semantic classification. The first is inspired by Mani et al., who rely on the tense of neighboring verbs to decide direction class. Since verb tense alone is inherently deictic, it is not sufficient to decide the direction, but we do add both the closest verb (w.r.t. dependency paths) and its POS tag (as well as any verbs directly related to this verb) as features. The second set of features compares day names, month names, and years to the document timestamp. The comparison determines whether, within a single cycle of the appropriate granularity (week for day-names and year for month-names), the point named by the timex would be before, after, or the same as the point referred to by the timestamp.

classifier	overall	after	before	same
BL	0.1749	0.4587	0.0802	0.1934
SVM	0.2245	0.4404	0.1578	0.2305
SVM_VERB	0.2094	0.3119	0.1631	0.2346
SVM_ALL	0.1185	0.2110	0.0989	0.1070
class dist	726	109	374	243

Table 4: Error rates: direction class.

We compare our learned classifier with a heuristic baseline classifier which first checks for the presence of a year or certain modifiers such as *ago* or *next* in the timex; if that fails, it computes the date features described above for each word in the timex and returns `same` if any word compares to the timestamp as same; if that fails, it uses the tense of the nearest verb; and finally, it defaults to `same`.

Table 4 shows the results of applying our classifiers to all phrase-aligned gold-standard point timexes. BL is the baseline; SVM, SVM_VERB, and SVM_ALL are the classifiers learned using our basic feature set, the basic feature set plus the verb features, and all the features, respectively. The learned classifier using all the features reduces the error rate of the baseline classifier by about a third. Note, though, that the learned classifiers without the date comparison features (SVM and SVM_VERB) perform substantially worse than even the baseline. One reason for this becomes clear from Table 5, which gives the error rates for the classifiers restricted to timexes consisting solely of a month or a day name. Unlike points in general, these timexes are all ambiguous with respect to direction and are, in fact, the primary motivation for both Mani et al. and Ahn et al. to consider direction classification as a separate task.

These results demonstrate that the date comparison feature is responsible for a substantial reduction in error rate (over 85% from SVM to SVM_ALL) and that for the `same` class, performance is perfect. This is largely due to the writing style of the documents, in which the current day is often referred to by name

classifier	overall	after	before	same
BL	0.1000	0.4348	0.1061	0.0000
SVM	0.3647	0.6087	0.3485	0.3086
SVM_VERB	0.3176	0.3478	0.3485	0.2840
SVM_ALL	0.0529	0.1304	0.0909	0.0000
class dist	170	23	66	81

Table 5: Error rates: direction month/day.

instead of as *today*, as in example (1).

Although both Mani et al. and Ahn et al. build direction classifiers, neither provide comparable results. Mani et al. do not evaluate their direction heuristics at all, and Ahn et al. train and test their machine learned classifier only on timexes determined to be ambiguous by their heuristics. In any case, their error rate is significantly higher, at 38%.

8 End-to-end performance

We now consider the performance of the entire system and the contributions of the components. First, though, we discuss our evaluation metrics.

8.1 Scoring

The official TERN scoring script computes precision and recall for VAL only with respect to correctly recognized TIMEX2s with a non-null VAL. While this may be useful in determining how far behind normalization is from recognition for a given system, it does not provide an accurate picture of end-to-end system performance, since the recall base does not include all possible timexes and the precision base does not include incorrectly recognized timexes.

The scoring script provides several raw counts that can be used to compute measures that are more indicative of end-to-end performance: `actTIMEX2` (# of actually recognized TIMEX2s); `corrTIMEX2` (# of correctly recognized TIMEX2s); `posVAL` (# of correctly recognized TIMEX2s with a non-null gold VAL); `corrVAL` (# of correctly recognized TIMEX2s with a non-null gold VAL for which the system assigns the correct VAL); and `spurVAL` (# of correctly recognized TIMEX2s with null gold VAL for which the system assigns a VAL). With these counts, we can define `corrNOVAL` (# of correctly recognized TIMEX2s with a null gold VAL for which the system assigns a null VAL), as $\texttt{corrTIMEX2} - \texttt{posVAL} - \texttt{spurVAL}$. We then define end-to-end precision (absP) and recall (absR) as $(\texttt{corrVAL} + \texttt{corrNOVAL})/\texttt{actTIMEX2}$ and $(\texttt{corrVAL}+\texttt{corrNOVAL})/\texttt{possTIMEX2}$, respectively. Official precision and recall for VAL are computed as $\texttt{corrVAL}/\texttt{actVAL}$ and $\texttt{corrVAL}/\texttt{possVAL}$.

8.2 Results

Our first set of results (Table 6(Top)), which are restricted to timexes in document TEXT elements, compares our system (LLL) to a version of our system (BL) that uses the baseline classifiers for semantic and direction class. It also presents a series of oracle results that demonstrate the effect of swapping in perfect classification for each of the learned classifiers. The oracle runs are labeled with a three-letter code in which the first letter ((P)erfect or (L)earned) refers to phrase classification; the second, to semantic classification; and the third, to direction classification. Note: perfect phrase classification is not the same as perfect recognition, since it excludes timexes that fail to align with parsed phrases.

Using the learned classifiers (LLL), which reduce error rates by about one-half for semantic class and one-third for direction class over the baseline classifiers, results in a five-point improvement in absolute F-measure over the baseline system (BL). We also see from runs LLP, LPL, and LPP that further improvement of these classifiers would substantially improve end-to-end performance. Finally, we see from runs PLL and PPP that recognition performance is a major limiting factor in our end-to-end scores.

In Table 6(Bottom), we present results over full documents, including metadata and text. LLL and PLL are the same as before; ITC-IRST is the system of (Negri and Marseglia, 2004), which achieved the highest official F-measure in the TERN 2004 evaluation. The results of our system (LLL) are comparable to those of ITC-irst: because we recognize fewer timexes, our official F-measure is higher (0.899 vs. 0.872) while our absolute F-measure is lower (0.769 vs. 0.806). We see from run PLL that our recognition module is largely to blame—with perfect phrase classification for recognition, our normalization modules produce substantially better results. With a system such as ITC-irst's, it is not possible to separate recognition performance from normalization performance, since there is a single rule base that jointly performs the two tasks—all normalizable timexes are presumably already recognized.

System	corrVAL	corrNOVAL	actTIMEX2	P	R	F	absP	absR	absF
BL	859	32	1245	0.813	0.787	0.800	0.716	0.624	0.667
LLL	931	33	1245	0.882	0.853	0.867	0.774	0.676	0.722
LLP	938	33	1245	0.912	0.859	0.885	0.780	0.680	0.727
LPL	951	39	1245	0.916	0.871	0.893	0.795	0.694	0.741
LPP	987	39	1245	0.951	0.904	0.927	0.824	0.719	0.768
PLL	1008	63	1287	0.886	0.828	0.856	0.832	0.751	0.789
PPP	1097	70	1287	0.966	0.901	0.932	0.907	0.818	0.860
LLL	1285	33	1601	0.910	0.887	0.899	0.823	0.721	0.769
PLL	1362	63	1643	0.912	0.866	0.888	0.867	0.780	0.821
ITC-IRST	1365	35	1648	0.875	0.870	0.872	0.850	0.766	0.806

Table 6: Performance on VAL. (Top): TEXT-only. (Bottom): full document.

9 Conclusion

We have described a novel architecture for a timex annotation system that eschews the complex set of hand-crafted rules that is a hallmark of other systems. Instead, we decouple recognition from normalization and factor out context-dependent semantic and pragmatic processing from context-independent semantic composition. Our architecture allows us to use machine learned classifiers to make context-dependent disambiguation decisions, which in turn allows us to use a small set of simple, context-independent rules for semantic composition. The normalization performance of this system is competitive with the state of the art and our overall performance is limited primarily by recognition performance. Improvement in semantic and direction classification will yield further improvements in overall performance. Our other plans for the future include experimenting with dependency relations for semantic composition instead of lexical patterns, evaluating our temporal anchor tracking method, and training the full system on other corpora and adapting it for other languages.

Acknowledgement This research was supported by the Netherlands Organization for Scientific Research (NWO) under project numbers 017.001.190, 220-80-001, 264-70-050, 354-20-005, 600.065.120, 612-13- 001, 612.000.106, 612.066.302, 612.069.006, 640.001.501, 640.002.501, and by the E.U. IST programme of the 6th FP for RTD under project MultiMATCH contract IST-033104.

References

D. Ahn, S. Fissaha Adafre, and M. de Rijke. 2005a. Extracting temporal information from open domain text: A comparative exploration. In R. van Zwol, editor, *Proc. DIR'05*.

D. Ahn, S. Fissaha Adafre, and M. de Rijke. 2005b. Recognizing and interpreting temporal expressions in open domain texts. In S. Artemov et al., editors, *We Will Show Them: Essays in Honour of Dov Gabbay, Vol 1*, pages 31–50.

S. Bethard and J.H. Martin. 2006. Identification of event mentions and their semantic class. In *Proc. EMNLP 2006*.

B. Boguraev and R. Kubota Ando. 2005. TimeML-compliant text analysis for temporal reasoning. In *Proc. IJCAI-05*.

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

E. Charniak. 2000. A maximum-entropy-inspired parser. In *Proc. NAACL 2000*, pages 132–139.

R. Dale and P. Mazur. 2006. Local semantics in the interpretation of temporal expressions. In *Proc. Workshop on Annotating and Reasoning about Time and Events*, pages 9–16.

L. Ferro, L. Gerber, I. Mani, and G. Wilson, 2004. *TIDES 2003 Std. for the Annotation of Temporal Expressions*. MITRE.

L. Ferro. 2004. Annotating the TERN corpus. `http://timex2.mitre.org/tern_2004/ferro2_TERN2004_annotation_sanitized.%pdf`.

K. Hacioglu, Y. Chen, and B. Douglas. 2005. Automatic time expression labeling for English and Chinese text. In A.F. Gelbukh, editor, *CICLing*, volume 3406 of *Lecture Notes in Computer Science*, pages 548–559.

J. Hitzeman. 1993. *Temporal Adverbials and the Syntax-Semantics Interface*. Ph.D. thesis, University of Rochester.

ISO. 1997. ISO 8601: Information interchange – representation of dates and times.

I. Mani and G. Wilson. 2000. Robust temporal processing of news. In *Proc. ACL'2000*, pages 69–76.

I. Mani, J. Pustejovsky, and R. Gaizauskas, editors. 2005. *The Language of Time: A Reader*. Oxford University Press.

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

E. Saquete, P. Martínez-Barco, and R. Muñoz. 2002. Recognizing and tagging temporal expressions in Spanish. In *Workshop on Annotation Standards for Temporal Information in Natural Language, LREC 2002*, pages 44–51.

F. Schilder. 2004. Extracting meaning from temporal nouns and temporal prepositions. *ACM TALIP*.

J. Wiebe, T. O'Hara, K. McKeever, and T. Öhrström Sandgren. 1997. An empirical approach to temporal reference resolution. In *Proceedings of EMNLP-97*, pages 174–186.