UTTime: Temporal Relation Classification using Deep Syntactic Features

Natsuda Laokulrat
The University of Tokyo
3-7-1 Hongo, Bunkyo-ku,
Tokyo, Japan
natsuda@logos.t.u-tokyo.ac.jp

Makoto Miwa
The University of Manchester
131 Princess Street,
Manchester, M1 7DN, UK
makoto.miwa@manchester.ac.uk

Yoshimasa Tsuruoka
The University of Tokyo
3-7-1 Hongo, Bunkyo-ku,
Tokyo, Japan
tsuruoka@logos.t.u-tokyo.ac.jp

Takashi Chikayama
The University of Tokyo
3-7-1 Hongo, Bunkyo-ku,
Tokyo, Japan
chikayama@logos.t.u-tokyo.ac.jp

Abstract

In this paper, we present a system, UTTime, which we submitted to TempEval-3 for Task C: Annotating temporal relations. The system uses logistic regression classifiers and exploits features extracted from a deep syntactic parser, including paths between event words in phrase structure trees and their path lengths, and paths between event words in predicate-argument structures and their subgraphs. UTTime achieved an F1 score of 34.9 based on the graphed-based evaluation for Task C (ranked 2^{nd}) and 56.45 for Task C-relation-only (ranked 1^{st}) in the TempEval-3 evaluation.

1 Introduction

Temporal annotation is the task of identifying temporal relationships between pairs of temporal entities, namely temporal expressions and events, within a piece of text. The temporal relationships are important to support other NLP applications such as textual entailment, document summarization, and question answering. The temporal annotation task consists of several subtasks, including temporal expression extraction, event extraction, and temporal link identification and relation classification.

In TempEval-3, there are three subtasks of the temporal annotation process offered, i.e., Task A: Temporal expression extraction and normalization, Task B: Event extraction, and Task C: Annotating temporal relations. This paper presents a system to handle Task C. Based on the annotated data provided, this subtask requires identifying pairs of temporal entities and classifying the pairs into one of the 14 relation types according to TimeML (Pustejovsky et al., 2005), i.e., *BEFORE, AFTER, IMMEDIATELY BEFORE, IMMEDIATELY AFTER, INCLUDES, IS INCLUDED, DURING, DURING INVERSE, SIMULTANEOUS, IDENTITY, BEGINS, BEGUN BY, END,* and *ENDED BY.*

The motivation behind our work is to utilize syntactic and semantic relationships between a pair of temporal entities in the temporal relation classification task, since we believe that these relationships convey the temporal relation. In addition to general features, which are easily extracted from sentences (e.g., part of speech tags, lemmas, synnonyms), we use features extracted using a deep syntactic parser. The features from the deep parser can be divided into two groups: features from phrase structure trees and features from predicate-argument structures. These features are only applicable in the case that the temporal entities appear in the same sentence, so we use only the general features for inter-sentence relations.

Predicate-argument structure expresses semantic relations between words. This information can be extracted from a deep syntactic parser. Features from predicate-argument structures can capture important temporal information (e.g., prepositions of time) from sentences effectively.

The remaining part of this paper is organized as follows. We explain our approach in detail in Section 2 and then show the evaluation and results in Section 3. Finally, we conclude with directions for future work in Section 4.

2 Approach

Our system, UTTime, is based on a supervised machine learning approach. UTTime performs two tasks; TLINK identification and classification. In

*Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 88–92, Atlanta, Georgia, June 14-15, 2013.

other words, UTTime identifies pairs of temporal entities and classifies these pairs into temporal relation types.

2.1 TLINK identification

A pair of temporal entities that have a temporal relation is called a TLINK. The system first determines which pairs of temporal entities are linked by using a ruled-based approach as a baseline approach.

All the TempEval-3's possible pairs of temporal entities are extracted by a set of simple rules; pairs of temporal entities that satisfy one of the following rules are considered as TLINKs.

- Event and document creation time
- Events in the same sentence
- Event and temporal expression in the same sentence
- Events in consecutive sentences

2.2 TLINK classification

Each TLINK is classified into a temporal relation type. We use a machine learning approach for the temporal relation classification. Two L2-regularized logistic regression classifiers, LIBLINEAR (Fan et al., 2008), are used; one for event-event TLINKs, and another one for event-time TLINKs. In addition to general features at different linguistic levels, features extracted by a deep syntactic parser are used.

The general features we employed are:

- Event and timex attributes

  All attributes associated with events (class, tense, aspect, modality, and polarity) and temporal expressions (type, value, functionInDocument, and temporalFunction) are used. For event-event TLINKs, we also use tense/class/aspect match, tense/class/aspect bigrams as features (Chambers et al., 2007).

- Morphosyntactic information

  Words, part of speech tags, lemmas within a window before/after event words are extracted using Stanford coreNLP (Stanford NLP Group, 2012).

- Lexical semantic information

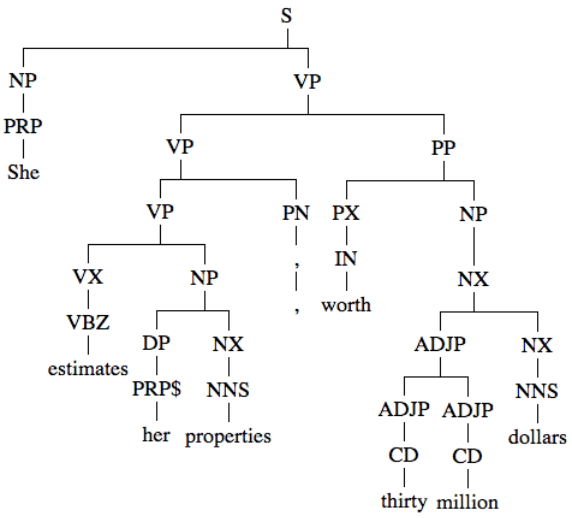

Figure 1: Phrase structure tree

  Synonyms of event word tokens from WordNet lexical database (Fellbaum, 1998) are used as features.

- Event-Event information

  For event-event TLINKs, we use *same_sentence* feature to differentiate pairs of events in the same sentence from pairs of events from different sentences (Chambers et al., 2007).

In the case that temporal entities of a particular TLINK are in the same sentence, we extract two new types of sentence-level semantic information from a deep syntactic parser. We use the Enju parser (Miyao and Tsujii, 2008). It analyzes syntactic/semantic structures of sentences and provides phrase structures and predicate-argument structures. The features we extract from the deep parser are

- Paths between event words in the phrase structure tree, and up(↑)/down(↓) lengths of paths.

  We use 3-grams of paths as features instead of full paths since these are too sparse. An example is shown in Figure 1. In this case, the path between the event words, *estimates* and *worth*, is *VBZ↑, VX↑, VP↑, VP↑, VP, PP↓, PX↓, IN↓*. The 3-grams of the path are, therefore, {*VBZ↑-VX↑-VP↑, VX↑-VP↑-VP↑, VP↑-VP↑-VP, VP↑-VP-PP↓, VP-PP↓-PX↓, PP↓-PX-↓-IN↓*}. The up/down path

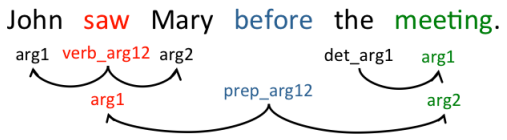

Figure 2: Predicate argument structure

lengths are 4 *(VBZ↑, VX↑, VP↑, VP↑)* and 3 *(PP↓, PX↓, IN↓)* respectively.

- Paths between event words in predicate-argument structure, and their subgraphs.

  For the previous example, we can express the relations in predicate-argument structure representation as

  - verb_arg12: estimate (she, properties)
  - prep_arg12: worth (estimate, dollars)

  In this case, the path between the event words, *estimates* and *worth*, is *←prep_arg12:arg1*. That is, the type of the predicate *worth* is *prep_arg12* and it has *estimate* as the first argument (arg1). The path from *estimate* to *worth* is in reverse direction (←).

  The next example sentence, *John saw mary before the meeting*, gives an idea of a more complex predicate-argument structure as shown in Figure 2. The path between the event words, *saw* and *meeting* is *←prep_arg12:arg1, prep_arg12:arg2*.

  We use (v, e, v) and (e, v, e) tuples of the edges and vertices on the path as features. For example, in Figure 2, the (v,e,v) tuples are *(see, ←prep_arg12:arg1, before)* and *(before, prep_arg12:arg2, meeting)*. In the same way, the (e,v,e) tuple is *(←prep_arg12:arg1, before, prep_arg12:arg2)*. The subgraphs of (v, e, v) and (e, v, e) tuples are also used, including *(see, ←prep_arg12:arg1, *), (*, ←prep_arg12:arg1, before), (*, ←prep_arg12:arg1, *), (*, prep_arg12:arg2, meeting), (before, prep_arg12:arg2, *), (*, prep_arg12:arg2, *), (*, before, prep_arg12:arg2), (←prep_arg12:arg1, before, *), (*, before, *)*.

From the above example, the features from predicate argument structure can properly capture the preposition *before*. It can also capture a preposition from a compound sentence such as *John met Mary before he went back home*. The path between the event words *met* and *went* are *(←conj_arg12:arg1, conj_arg12:arg2)* and the (v, e, v) and (e, v, e) tuples are *(met, ←conj_arg12:arg1, before)*, *(before, conj_arg12:arg2, went)*, and *(←prep_arg12:arg1, before, prep_arg12:arg2)*.

2.3 Hybrid approach

The rule-based approach described in Section 2.1 produces many unreasonable and excessive links. We thus use a machine learning approach to filter out those unreasonable links by training the model in Section 2.2 with an additional relation type, *UNKNOWN*, for links that satisfy the rules in Section 2.1 but do not appear in the training data.

In this way, for Task C, we first extract all the links that satisfy the rules and classify the relation types of those links. After classifying temporal relations, we remove the links that are classified as *UNKNOWN*.

3 Evaluation

The scores are calculated by the graph-based evaluation metric proposed by UzZaman and Allen (2011). We trained the models with TimeBank and AQUAINT corpora. We also trained our models on the training set with inverse relations. The performance analysis is based on 10-fold cross validation on the development data.

3.1 Task C

In Task C, a system has to identify appropriate temporal links and to classify each link into one temporal relation type. For Task C evaluation, we compare the results of the models trained with and without the features from the deep parser. The results are shown in Table 1. The rule-based approach gives a very low precision.

3.2 Task C-relation-only

Task C-relation-only provides a system with all the appropriate temporal links and only needs the system to classify the relation types. Since our goal is to exploit the features from the deep parser, in Task C-relation-only, we measured the contribution of those features to temporal relation classification in Table 2.

Features	F1	P	R
gen. (rule)	22.51	14.32	52.58
gen. + ph. + pas. (rule)	22.61	14.30	54.01
gen. + ph. + pas. (hyb.)	33.52	36.23	31.19
gen. + ph. + pas. (hyb. + inv.)	39.53	37.56	41.70

Table 1: Result of Task C. (rule: rule-based approach, hyb.: hybrid approach, gen.: general features, ph.:phrase structure tree features, pas.:predicate-argument structure features, and inv.: Inverse relations are used for training.)

Features	F1	P	R
gen.	64.42	64.59	64.25
gen. + ph.	65.24	65.42	65.06
gen. + pas.	66.40	66.55	66.25
gen. + ph. + pas.	66.39	66.55	66.23
gen. + ph. + pas. (inv.)	65.30	65.39	65.20

Table 2: Result of Task C-relation-only. (gen.: general features, ph.:phrase structure tree features, pas.:predicate-argument structure features, and inv.: Inverse relations are used for training.)

The predicate-argument-structure features contributed to the improvement more than those of phrase structures in both precision and recall. The reason is probably that the features from phrase structures that we used did not imply a temporal relation of events in the sentence. For instance, the sentence "*John **saw** Mary before the **meeting***" gives exactly the same path as of the sentence "*John **saw** Mary after the **meeting***".

3.3 Results on test data

Tables 3 and 4 show the results on the test data, which were manually annotated and provided by the TempEval-3 organizer. We also show the scores of the other systems in the tables. For the evaluation on the test data, we used the models trained with general features, phrase structure tree features, and predicate-argument structure features.

UTTime-5 ranked 2^{nd} best in Task C. Interestingly, training the models with inverse relations improved the system only when using the hybrid approach. This means that the inverse relations did not improve the temporal classification but helped the system filter out unreasonable links (UNKNOWN) in the hybrid approach. As expected, the ruled-based approach got a very high recall score at the expense of precision. UTTime-1, although it achieved the F1 score of only 24.65, got the highest recall among all the systems.

Approach	F1	P	R
rule (UTTime-1)	24.65	15.18	65.64
rule + inv (UTTime-3)	24.28	15.1	61.99
hyb. (UTTime-4)	28.81	37.41	23.43
hyb. + inv. (UTTime-5)	34.9	35.94	33.92
cleartk	**36.26**	**37.32**	**35.25**
NavyTime	31.06	35.48	27.62
JU-CSE	26.41	21.04	35.47
KUL-KULTaskC	24.83	23.35	26.52

Table 3: Result of Tack C on test data. (rule: rule-based approach, hyb.: hybrid approach, and inv.: Inverse relations are used for training.)

Approach	F1	P	R
gen. + ph. + pas. (UTTime-1)	**56.45**	**55.58**	**57.35**
gen. + ph. + pas. (UTTime-2)	54.26	53.2	55.36
gen. + ph. + pas. (inv.) (UTTime-3)	54.7	53.85	55.58
NavyTime	46.83	46.59	47.07
JU-CSE	34.77	35.07	34.48

Table 4: Result of Task C-relation-only on test data. (gen.: general features, ph.:phrase structure tree features, pas.:predicate-argument structure features, and inv.: Inverse relations are used for training.)

For Task C-relation-only, we achieved the highest F1 score, precision, and recall. UTTime-2 basically had the same models as that of UTTime-1, but we put different weights for each relation type. The results show that using the weights did not improve the score in graph-based evaluation.

4 Conclusion

The system, UTTime, identifying temporal links and classifying temporal relation, is proposed. The links were identified based on the rule-based approach and then some links were filtered out by a classifier. The filtering helped improve the system considerably. For the relation classification task, the features extracted from phrase structures and predicate-argument structures were proposed, and the features improved the classification in precision, recall, and F-score.

In future work, we hope to improve the classification performance by constructing timegraphs (Miller and Schubert, 1999), so that the system can use information from neighbor TLINKs as features.

References

James Pustejovsky, Robert Ingria, Roser Saurí, José Castaño, Jessica Littman, Rob Gaizauskas, Andrea Setzer, Graham Katz, Inderjeet Mani 2005. The specification language TimeML. *The Language of Time: A reader*, pages 545–557

Stanford Natural Language Processing Group. 2012. Stanford CoreNLP.

Christiane Fellbaum. 1998. WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification.

Nathanael Chambers, Shan Wang and Dan Jurafsky. 2007. Classifying Temporal Relations between Events. In *ACL 2007*, pages 173–176.

Yusuke Miyao and Jun'ichi Tsujii. 2008. Feature Forest Models for Probabilistic HPSG Parsing. In *Computational Linguistics.* 34(1). pages 35–80, MIT Press.

Naushad UzZaman and James F. Allen. 2011. Temporal Evaluation. In *ACL 2011*, pages 351–356.

Stephanie A. Miller and Lenhart K. Schubert. 1999. Time Revisited. In *Computational Intelligence 6*, pages 108–118.