Representation and Learning of Temporal Relations

Leon Derczynski
Department of Computer Science
University of Sheffield
S1 4DP, UK
leon.d@shef.ac.uk

Abstract

Determining the relative order of events and times described in text is an important problem in natural language processing. It is also a difficult one: general state-of-the-art performance has been stuck at a relatively low ceiling for years. We investigate the representation of temporal relations, and empirically evaluate the effect that various temporal relation representations have on machine learning performance. While machine learning performance decreases with increased representational expressiveness, not all representation simplifications have equal impact.

1 Introduction

Textual accounts often contain descriptions of events, times, and how they relate to one another temporally. To connect events and times to each other, we need to know the kind of temporal ordering between them. This ordering can be modeled with **temporal relations** that hold between pairs of entities, each of which may be an event or time. Extracting these relations is critical to understanding the text: for example, given an almanac of presidents of the USA, there are likely to be many indications of different people being president – but only one will be factual at any given time. In order to reason about events and the applicability of information in a document, linguistic expressions of time need to be converted to a formal representation. This task is temporal relation annotation.

To annotate temporal relations for reasoning or information extraction, one must to select a way of representing temporal relations. Such representations typically comprise a **set of temporal relations**, with each member describing a different temporal ordering. Building such representations is a key artificial intelligence task in reasoning and planning, and proposed solutions have amounted to major work in the field; e.g. Allen (1984), Freksa (1992).

Temporal relation annotation has two key parts. One must decide which entity pairs to relate, and then determine the nature of their relation. These are referred to as temporal relation **identification** and temporal relation **typing**.

These may be approached as a joint task, especially when it is possible for the type of one link to influence the type of another. For example, if event A is before event B and that event B is before event C, due to transitivity, the choice of relations between A and C may be constrained. Choosing how to represent the types of temporal relations is the focus of this paper.

Machine learning of temporal relations is hard. Mani et al. (2007) detail experiments with features annotated in TimeML (Pustejovsky et al., 2004), and reach around 75% accuracy at overall relation typing,[1] using gold-standard relation identification.[2] This pattern repeats in subsequent literature. Many others reach a similar performance, or perhaps even exceed it by 1-2% (Mirroshandel et al., 2011; Do et al., 2012); some do well by focusing on specific sub-parts of the problem (e.g. relations that are expressed by tense shifts (Derczynski and Gaizauskas, 2013b); relations between events in the same verb clause construction (Bethard et al., 2007)), or in contrast by taking a holistic whole-graph approach (Chambers

[1] Calculated by weighing the event-event and event-timex relation scores in their paper according to the distribution in the dataset used.

[2] This is typically made up of around 70% for event-event relations and 80% for event-time relations.

Relation	Symbol	Explanation of A-relation-B
before	<	A finishes before B starts
after	>	A starts after B ends
equals	=	A and B happen at the same time
meets	m	A happens immediately before B
is met by	mi	A happens immediately after B
overlaps	o	A is an interval during which B starts but does not finish
is overlapped by	oi	A starts within B but finishes after
during	d	A happens between B's start and finish
contains	di	A starts before, and finishes after, B
starts	s	A and B start at the same time, but B continues past A's end
is started by	si	A starts at the same time as B, but then goes on for longer
finishes	f	A starts after B, but they finish at the same time
is finished by	fi	A starts before B, and they finish at the same time

Table 1: Temporal interval relations, using Allen's symbols

et al., 2014); or relations where a temporal conjunction is present (Derczynski and Gaizauskas, 2013a)); but no general breakthrough appears to be on the horizon.

Indeed, the top systems in each of the TempEval challenges had tightly-clustered scores showing some 5-10% error reduction over the most-common-class baseline, despite the work being spread over years of active research (Verhagen et al., 2009; UzZaman et al., 2013; Bethard et al., 2016).

When one considers how human annotators cope with the task as it is often cast, it is unsurprising that machine performance is not high. Choosing from a set of abstract temporal relation types to fit an arbitrary pair of event mentions in a given document is difficult. For the largest TimeML corpus, inter-annotator agreement on relation types was just 0.71 (kappa) between a set of experts familiar with temporal annotation.[3] In this case, they had to assign one of a set of fourteen different temporal ordering types to the pair of events.

Problems may lie in the schema of temporal relation types. Assigning types requires annotators to perform abstract reasoning often with incomplete information; systems must do the same. Thus, the choice of representation for temporal relation types is critical. The more relation types to choose from, the more reasoning is required, and so the potential for error increases.

The bulk of work has concentrated on describing the order of two events by using relations from Allen's interval algebra. However, no work has directly investigated the effect of temporal relation representation design on performance of machine learning systems.

This paper investigates temporal relation type representations, in the framework of TimeML when possible. It compares a range of existing relation sets and discuss two dimensions for relation representation design. Following this is a general comparison of the impact that relation set choice can have on machine learning performance. Next follows discussion of techniques commonly used to improve training sets, including relation type mapping, and examine their impact. Finally, the paper investigates how discriminable often-conflated relations are, providing insight into the difficulty added by using finer-grained relation sets, and reports on research on human aspects of temporal relation representation.

2 Temporal Representation Schemes

This section introduces fundamental temporal relation type sets for linguistic annotation. Standards for representing temporal relations are then described in the context of these fundamental relation sets.

[3] See http://timeml.org/site/timebank/documentation-1.2.html

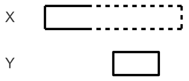
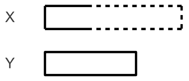
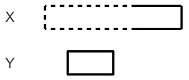
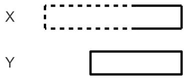
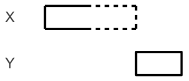
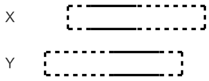
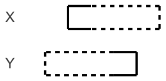
Relation	Illustration	TimeML relation type disjunction
X is *older* than Y Y is *younger* than X		X [BEFORE, IBEFORE, ENDED_BY, INCLUDES, DURING] Y
X is *head to head* with Y		X [BEGINS, SIMULTANEOUS, IDENTITY, BEGUN_BY] Y
X *survives* Y Y is *survived by* X		X [INCLUDES, BEGUN_BY, IAFTER, AFTER] Y
X is *tail to tail* with Y		X [ENDED_BY, SIMULTANEOUS, IDENTITY, ENDS] Y
X *precedes* Y Y *succeeds* X		X [BEFORE, IBEFORE, ENDED_BY, INCLUDES, DURING_INV] Y
X is a *contemporary* of Y		X [INCLUDES, IS_INCLUDED, BEGUN_BY, BEGINS, DURING, DURING_INV, SIMULTANEOUS, IDENTITY, ENDS, ENDED_BY] Y
X is *born before death* of Y Y *dies after birth* of X		X [IS_INCLUDED, ENDS, DURING_INV, BEFORE, IBEFORE, INCLUDES, DURING, ENDED_BY] Y

Table 2: Semi-interval relations. Adapted from Freksa (1992). The superset of relations is omitted here, but related in that work.

2.1 A Very Simple Relation Set

A Very Simple set of relations for representing temporal relations could have just three members: BEFORE, OVERLAP, and AFTER. These are mutually exclusive; before means wholly before, and after means wholly after, precluding any ambiguity once a relation type has been assigned. No specific model of events is required here.

However, this representation is too simplistic to describe many of the temporal relations that are often explicitly conveyed in language. For example, there is no way to represent the distinction between *after* and *just after*, or between *I opened the door yesterday* and *I held the door open all day yesterday*.

2.2 Interval Relations

Allen (1983) introduces a richer relation set which models the events (or times) that are related as **temporal intervals**. Each interval consists of a start and end point, which correspond to when the item in question began and ended. Based on this, thirteen possible interval arrangements are identified and given names. The relations are shown in Table 1. This model can be used to model relations between events and times, both those observed and those described in natural language (Allen, 1984).

2.3 Semi-Interval Relations

Freksa (1992) describes a relation set based on semi-intervals, with temporal reasoning based on natural language as one of its intended uses. In this semi-interval relation set and algebra, events and times are still modelled as temporal intervals, but one is not required to fully specify the bounds of both related intervals in order to describe the relation between them. Rather, different relation types apply depending on both the order and the degree of specification. This allows for a more expressive representation. A subset of these underspecified interval relations are demonstrated in Table 2.

Its relation types (or "conceptual neighbourhoods") can be thought of as disjunctions of Allen interval relation types. For example, Freksa's TAIL-TO-TAIL relation corresponds to any situation where both intervals end simultaneously (in Allen's terms, $\mathtt{fi} \vee = \vee \mathtt{f}$). The CONTEMPORARY relation is analogous to our Very Simple relation set's OVERLAP type (Section 2.1). If A is the set of Allen interval relation types and F the set of Freksa semi-interval relation types, then $F \subset \mathcal{P}(A)$. This semi-interval relation set is flexible, but this comes at a cost. Annotators and relation typing systems have a greater selection of relation types to choose from: there are total 31 types.

These three representations provide most of the concepts required to understand the other major relation type sets.

2.4 Annotation Standards

TimeML TimeML uses the Allen relations (although slightly renamed), and adds a IDENTITY relation to indicate co-reference. This extra relation is temporally equivalent to SIMULTANEOUS. Interpretation of TimeML's DURING and DURING_INV relations has been ambiguous; this work adopts the TimeML-strict definitions (Derczynski et al., 2013), which map directly to Allen's overlap and overlap-inverse.

As with Allen's temporal interval relations, TimeML requires full specification of both intervals and established versions only permit assignment of a single relation type. Links in newer prototype versions of ISO-TimeML (Pustejovsky et al., 2010) permit selection of disjunctions of interval relation types.

TempEval TempEval-1 and TempEval-2 (Verhagen et al., 2009; Verhagen et al., 2010) used the same three types as the Very Simple scheme, plus two less-specific types – BEFORE-OR-OVERLAP and AFTER-OR-OVERLAP – and the relaxed VAGUE. TempEval-3 used TimeML relations.

STAG The STAG annotation scheme (Setzer and Gaizauskas, 2000) – a precursor to TimeML – specifies three basic relations: SIMULTANEOUS, BEFORE and INCLUDES. Representation of "after" and "is-included-by" relations can be achieved by swapping the order of arguments (i.e. A BEFORE B vs. B BEFORE A). This intuitive, simple scheme cannot express Allen-style overlap, and the STAG INCLUDES relation is vague in that it subsumes BEGINS and STARTS Allen relation types.

Narrative Containers Pustejovsky and Stubbs (2011) attempt to reduce the scope of temporal relation typing task by defining narrative containers over sets of event mentions in documents. These containers represent a time interval during which groups of events occur. The containing time interval may either be explicit and reified with a narrative time, or implicit, with the exact bounds determined by context and text type. Containers reduce the amount of uncertainty in relation typing by grouping related events into local contiguous temporal scopes. This reduces annotator load and provides a basis for the temporal structure of a given document, without requiring specification of the type of every temporal relation. The narrative container approach is designed to support event-to-time relation typing, and succeeds in doing so directly for 50% of events with more informative relation descriptions. Empirical work (Miller et al., 2013) indicates that these containers can be annotated by humans sufficiently well to learn an initial automated approach. However, in the absence of a large general-purpose corpus annotated with them, this paper does not include narrative containers.

For a general overview and history of temporal annotation standards, see (Strötgen and Gertz, 2016).

3 Analysing temporal relation sets

This section investigates selected sets of temporal relation types. The sets are chosen to demonstrate key differences, though the list is not exhaustive. A graph-based comparison of three sets can also be found in (Denis and Muller, 2010).

3.1 Expressiveness and Specificity

We qualitatively describe temporal relation sets in two dimensions: The first dimension, *expressiveness vs. simplicity* details the range of different combinations of event orderings they can capture. The second, *specificity vs. laxness* details how much constraint the relation set's types imply, and how much one needs to know before typing a relation.

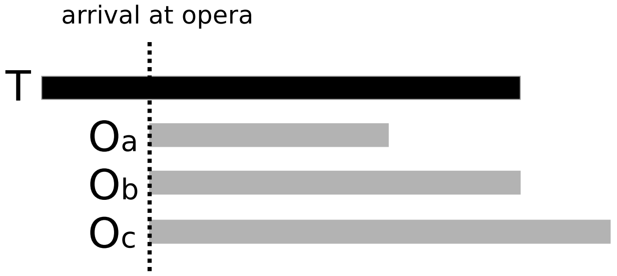


Figure 1: The Opera problem: Linguistic ambiguity leads to inability to choose a single interval relation. Here, we know the time of arrival at the opera but not of the departure, making it hard to relate intervals O and T.

For example, the Very Simple relation set is not very expressive – it has only three relations. The interval set is specific – one must describe relations using both endpoints of both intervals. Compared to the interval set, semi-interval relations are more expressive and less specific, because less information is required to choose a relation type.

3.2 Precision and Annotation

When describing the order of events and times, it is important to get the right degree of precision. Time can be thought of as a unidirectional, continuous dimension. This continuous aspect of time permits relatively small differences in the alignments of events, which may be at best useless and at worst counterproductive when reasoning.

However, lacking an inherent temporal quantisation or discretisation framework (such as chronons (Lévi, 1927)), we have no objective ground for ignoring small discrepancies in event timing. Instead, cues come from the temporal scale at which events take place (Gaizauskas et al., 2012).

A pragmatic approach suits this situation. Annotations over a text should match the information expressed. So, in *"I smiled when she entered the room"*, whether the smiling and the entrance were delayed by a few fractions of a second is unlikely to be salient. Rather, the language indicates that these events happened at the same time, and so this is the best thing to annotate. Texts regarding quantum events or star formation would both use different temporal scales.

3.3 The Opera Problem

The interval-based relation set is a richer representation than the Very Simple one, but has the disadvantage that the order of both related intervals' start and end (i.e. four points in total) must be known before a single relation type can be chosen. It is desirable to choose a single relation type when annotating linguistic data in order to keep down the number of choices that annotators need to make, which in turn affects the quality of output. The linguistic ambiguity in natural language texts can make it hard to choose a single interval relation type. This can be illustrated using the Opera problem, as put forward in Derczynski (2017)

Given the statement *Irene went to the opera today*, assume interval O is the visit to the opera and interval T is today.[4] Without knowing when Irene left, one cannot choose a single interval relation out of (a) O is included in T, (b) O and T end at the same time, or (c) O overlaps T – see Figure 1.

Underspecifying the endpoints of intervals allows ambiguity to be captured in a single relation type. The semi-interval relation set offers a solution the Opera problem using a relation where the start of O is specified but its end is not. In this representation it is O YOUNGER-CONTEMPORARY-OF T, which corresponds to the disjunction of Allen relations $\texttt{f} \vee \texttt{d} \vee \texttt{oi}$.

Similarly, the Very Simple set can give a lossy representation for this problem with its OVERLAP relation. It is lossy because the text describes more than can be represented by this relation type.

Confusions like the one the Opera problem presents occur frequently in TimeML annotation when linking events to the document time stamp, which is typically a one-day interval. For example, news

[4] This treats O as an eventuality which represents the state of Irene being at the opera; (Hobbs and Pustejovsky, 2003) examines the semantics of events in the context of temporal interval representations.

stories – the bulk of the TimeML annotated data in existence – refer to the day in which they are reported, but as one cannot see into the future and determine whether or not assertions and events in the news will persist past the end of the day, it is not always possible to accurately assign a single relation type with certainty.

4 Machine Learning and Temporal Relation Types

There are specific problems when applying machine learning to the temporal relation typing task. When approached as a classification task, the assumption made by most tools is that classes are independent. However, this is not correct in this case, for two reasons.

Firstly, relation types are not all equally different. TimeML's "immediately after" relation, IAFTER, is more like AFTER than it is INCLUDES. In fact, this non-orthogonality is critical to the construction and behaviour of Freksa's conceptual neighbourhood relations. The connected nature of relation types offers nuance in training, classification and evaluation of machine learning approaches to temporal relation typing.

Secondly, relation typing is a highly structured task, with local and global dependencies. Not only should local relation constraints (e.g. through transitivity and commutativity) be taken into account, as noted by Hovy et al. (2012); also, the annotation of a well-formed document should be globally temporally consistent, otherwise it will detail an impossible sequence of events.[5]

This interdependent aspect of temporal relations is hard to model, partially due to the underlying computational complexity (Vilain and Kautz, 1986). Nevertheless, including features for local dependencies give modest accuracy improvements (Chambers and Jurafsky, 2008; Yoshikawa et al., 2009).

5 Evaluating Learning of Relation Sets

Using a variety of temporal relation sets, machine learning of temporal relations is evaluated over the same data. This is achieved by mapping TimeML gold-standard relations to other relation sets thus:

Simple – before: BEFORE, IBEFORE; after: AFTER, IAFTER; overlap: everything else.

STAG – add inverses of the before and includes types, to make five relations. before: BEFORE, IBEFORE; after: AFTER, IAFTER; simultaneous: SIMULTANEOUS, DURING, DURING_INV, IDENTITY; is-included: IS-INCLUDED; includes: everything else.

Allen – One change: TimeML IDENTITY becomes Allen's "=" (equal, i.e. simultaneous).

TimeBank 1.2 (Pustejovsky et al., 2003) is the corpus, which has 6 418 temporal relations (TimeML TLINKs) in 183 documents. This is merged with the AQUAINT TimeML corpus, which has 2 340 TLINKs in 73 documents.[6] The AQUAINT corpus is structured in four themes, each of which contains multiple documents tracking the same story over time.

A 75%/25% split training and evaluation split is used. To avoid information that could be extracted through inference leaking between documents, splits are made at document and not TLINK level.

When evaluating learning of temporal information, one must also be wary of **timeline pollution** (Bergmeir and Benítez, 2012) – that is, introducing later knowledge into the training set that might give clues as to the contents of an earlier document in the test set. This pollution concern is offset by avoiding duplicating text across splits. These factors work against each other: avoiding timeline pollution, by only including later-dated documents in the test split, increases the risk of these documents re-using fragments of earlier articles on the same topic. Avoiding re-use is probably more important from a basic NLP point of view, so splits are arranged to reduce this. The solution adopted in the AQUAINT corpus is to split at theme-level, as a lot of similar text is repeated across each theme.

The feature set is the same as that in Mani et al. (2007). These are the attributes and values of the TimeML annotations, plus the strings they annotate, and in addition: the string of the conjunction coordinating the relation, if present; and two flags indicating whether, in a relation between two verb events, the verbs have the same tense or same aspect. Following previous work (Mani et al., 2007), experiments use a maximum entropy classifier (MegaM (Daumé III, 2004)), which has consistently provided stable

[5] This requirement makes the assumption that the author has written a temporally consistent story.

[6] From http://timeml.org/site/timebank/timebank.html

Relation set	MCC baseline	MaxEnt	ID3
Simple	44.79	73.96	57.65
STAG	38.00	60.36	51.13
Allen	26.08	58.58	46.01
TimeML	26.08	57.08	45.51

Table 3: Relation typing precision using a variety of temporal relation sets, compared to a most-common-class baseline.

Relation set	MCC baseline	MaxEnt	ID3
TempEval-2	55.61	59.25	61.73
TimeML	33.87	50.43	33.03

Table 4: Relation typing precision on a smaller set of links, using the TempEval-2 relations.

results in this task, and a decision tree classifier, ID3 (Quinlan, 1986), for its different inductive bias – i.e., away from the independence assumption. The goal here is to highlight relative performance using various relation representations. Results are given in Table 3.

Some relation sets could not be directly mapped from TimeBank. The TempEval-2 data uses a subset of TimeBank, but due to its relation set's multiple levels of specification, cannot be directly mapped. In a second set of experiments, the TempEval-2 dataset is compared with the corresponding TimeML relations. Performance using TimeML is included for comparison. Results are given in Table 4.

Generally, results suggest automatic relation typing is easier on smaller and simpler relation type sets. Greater expressiveness leads to both lower baselines and lower ML performance. Note the performance difference between Allen and TimeML relations sets – these are identical except for the addition of a co-reference relation in TimeML, for which relations are slightly harder to learn. Our Very Simple relation set of just three coarse-grained temporal arrangements was the easiest to assign using the existing data.This anti-monotone relation between increased expressiveness and lower machine learning performance held over all representations and algorithms tested.

6 Adapting Training Data

This section examines techniques for improving machine learning of temporal relations: simplifying the problem, and increasing available training data. Taking transitive closures is omitted, as it does not help (Mani et al., 2007).

6.1 Folding

Many relation types used in both TimeML and Allen's relation sets have a corresponding inverse relation. Mapping between these can be achieved by inverting the relation type and the argument order. For example, BEFORE*(Monday, Tuesday)* is equivalent to AFTER*(Tuesday, Monday)*. This relation **folding** simplifies classification by reducing the number of target classes, and is common in temporal relation classification, e.g. Mani et al. (2007); Mirroshandel et al. (2011).

6.2 Doubling

Doubling the reverse of folding: instead of using a relation type mapping to reduce the number of classes, use the mappings to double the number of examples. In controlled evaluation, it is possible to reverse the order of arguments in the evaluation set so the set only contains relation types the classifier has seen in folded training data. This is not possible where the relation type is never known, as one does not have control over argument order. E.g. if all AFTER relations are removed from the training data by swapping their arguments and changing them to BEFORE, when faced with the previously-unseen relation C AFTER D, a classifier trained on folded data will fail.

To solve this, instead of reducing the set of available relations, one can increase training data size by using the folding mappings to create new training instances instead of altering existing ones. That is, given an example A AFTER B one automatically adds an extra B BEFORE A. This is relation doubling.

Data set	MCC baseline	MaxEnt	ID3
Unfolded TimeML	26.08	57.08	45.51
Doubled TimeML	26.08	28.68	18.58
Folded TimeML	42.12	70.48	59.93

Table 5: Relation typing precision on a smaller set of links, using folding and doubling.

6.3 Closure

Many relation sets have types that exhibit properties of transitivity or commutativity. While a useful technique for increasing the volume of available data, the results of adding training data generated through closure can be unpredictable, and it is not always a suitable technique to apply; therefore, we omit effects of temporal graph closure here.

For example, in the interval relations, if A BEFORE B and B BEFORE C, then by transitive closure A BEFORE C. These properties allow the automatic addition of temporal relations to an existing set through inference. The set of links resulting from inferring as many possible relations from a source document is called that document's temporal closure.

While a useful technique for increasing the volume of available data, the results of adding training data generated through closure can be unpredictable, and it is not always a suitable technique to apply. For further reference, results on the use of closed data are presented by Mani et al.; extended discussion can be found in Verhagen (Verhagen, 2004).

6.4 Evaluation

Machine learning performance on folded training data using the TimeML relation set is evaluated as above. Results are in Table 5. Folding offers a big improvement over other methods, in terms of both error reduction above most-common-class baseline and also absolute accuracy. However, it is not applicable to real-world data. While doubling overcomes this critical deficiency, it overall reduces performance.

7 Relation Distinctions

As relation sets become more complex, categories of temporal relation are subdivided. For example, TimeML distinguishes between immediately before, IBEFORE, and before with an interceding gap, BEFORE, whereas neither our simple set, the STAG set or the TempEval-2 set do. It may be that some of these distinctions are easier to learn than others. To investigate, subgroups of relations that may be (or have been) sources of potential confusion are identified, and classifiers trained just on relations of the types in the group. These were then evaluated on the relations in the group from the test examples. As these subdivisions result in small amounts of training data, folded relations are used to simplify the problem and improve the examples to classes ratio.

The confusion between TimeML DURING and INCLUDES relations is also investigated. These were identified by annotators as confusing during TempEval-3 and cursory examination during TempEval-3 revealed many questionably annotated examples in TimeBank.

Results are shown in Table 6. Many of the finer distinctions can be successfully made in these situations, though there is significant variation. Unsurprisingly, accuracy on each relation type is higher the more examples are seen, with most having accuracies above 80% and some in the high 90s. The *overlap* group was hardest, at an accuracy of 55.23%. It also has one of the least skewed class distributions. It is interesting to note that relations in the STAG *includes* group, while sharing much with the *overlap* group, was generally easier to distinguish from one another. The co-reference relation added to the Allen relations by TimeML is also relatively easy to distinguish from non co-referent simultaneity, with the classifier reaching 30% – 50% error reduction above baseline. Indeed, (Glavaš and Šnajder, 2013) achieve F1 scores as high as 0.95 using graph kernels on this problem. With regard to the TimeML INCLUDES/DURING confusion, binary classification accuracy is lower than the most-common-class baseline; an indication of a difficult problem, or of poor-quality data – which uncertain annotators would generate.

Coarse relation	TimeML rel'n	Data %	Acc.
Before (3 300 instances)	BEFORE	97.17	99.65
	IBEFORE	2.83	31.34
	Overall	-	97.72
After (1 590 instances)	AFTER	95.34	99.12
	IAFTER	4.66	20.00
	Overall	-	95.43
Overlap (2 792 instances)	IDENTITY	35.52	64.98
	INCLUDES	31.09	61.29
	SIMUL.	23.89	50.82
	ENDS	4.91	47.45
	BEGINS	4.12	11.30
	DURING	3.47	3.09
	Overall	-	55.23
STAG includes (1 120 instances)	INCLUDES	77.50	96.20
	ENDS	12.23	52.55
	BEGINS	10.27	19.13
	Overall	-	82.95
Coreference distinction (1 646 instances)	IDENTITY	57.65	78.06
	SIMUL.	42.35	60.87
	Overall	-	72.76
TimeML "during" (4 284 instances)	INCLUDES	89.95	97.70
	DURING	10.05	11.34
	Overall	-	89.02

Table 6: Folded relation typing precision when training on and classifying subgroups of similar TimeML relation types.

8 Human Factors in Temporal Annotation

When it comes to designing new representations, human factors can have bearing on design choices. Scheuermann et al. (2013) provide valuable insights into annotator preferences in temporal annotation. They report that experts prefer more expressive representations, at the cost of simplicity, whereas non-experts lean towards perceived user-friendliness at the cost of expressivity. This matches the anti-monotone interaction between machine learning performance and representational expressiveness: the simpler the representation, the easier it is to annotate and to learn.[7] Large variation was seen across all annotators. Schaeken and Johnson-Laird (2000) show that agreement and success of humans annotating timelines can be perturbed by even slight changes in text phrasing, which perhaps explains the variations seen in Scheuermann et al.'s work.

These findings are worth considering in the construction of relation representations and human temporal annotation tasks. One caveat, as Allen (1991) points out, is that the presence of parallels between natural language and a given representation is not always indicative of its suitability for automated reasoning. Freksa's relation set may be a better target for human annotation: it is intended to better reflect the usage of temporal relation expression in natural language, and only requires judgment over pairs of points rather than intervals, for construction. For example, one may decompose interval relations into point relations, and ask "Does A begin before B ends?". We already know that simpler questions yield better results from annotators (Sabou et al., 2014), and so a decomposed, less constrained approach to annotation may be fruitful. Certainly, the 13-way Allen (or 14-way TimeML) task has been tough, and it is reasonable that some disagreements come from annotators having to chose a particular relation type when in fact many (or none) seem appropriate to them.

9 Conclusion

This paper has investigated choices in representation of temporal relations. Human annotation agreement is often low in relation typing, and the task is also difficult for automatic systems. The expressivity of a representation has an inverse correlation with automatic temporal relation typing performance.

However, more expressive representations are required in order to accurately capture temporal structure, or even to reduce annotator confusion. It was shown that decisions over fine distinctions between

[7] The effect agreement has on training data is not measured here, but is stable across sets at around kappa 0.7.

temporal relation types in more expressive systems are of varying difficulty. This finding fits the observation that not all temporal relation types are equally different from each other.

In summary, expressivity is lacking in popular temporal relation representations, but large relation types sets are harder to annotate. It is important to get specificity right in a given relation set, and to select carefully the groupings made when creating coarser relation sets. Therefore, being aware of the expressiveness of different representations is critical to choosing how to represent temporal relations effectively for a given scenario.

Acknowledgements

Thanks to all who reviewed this paper, for their extensive, constructive feedback. Thanks also to Rob Gaizauskas for feedback on precursors to this work, and to Aarhus University for their kind facility provision. This research has received support from the EU from the H2020 program under grant agreement No. 687847, COMRADES.

References

J. F. Allen. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

J. F. Allen. 1984. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154.

J. F. Allen. 1991. Time and time again: The many ways to represent time. *International Journal of Intelligent Systems*, 6(4):341–355.

C. Bergmeir and J. M. Benítez. 2012. On the use of cross-validation for time series predictor evaluation. *Information Sciences*, 191:192–213.

S. Bethard, J. Martin, and S. Klingenstein. 2007. Timelines from Text: Identification of Syntactic Temporal Relations. In *Proceedings of the International Conference on Semantic Computing (ICSC)*, pages 11–18.

S. Bethard, G. Savova, W.-T. Chen, L. Derczynski, J. Pustejovsky, and M. Verhagen. 2016. Semeval-2016 task 12: Clinical TempEval. *Proceedings of SemEval*, pages 1052–1062.

N. Chambers and D. Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 698–706. ACL.

N. Chambers, T. Cassidy, B. McDowell, and S. Bethard. 2014. Dense Event Ordering with a Multi-Pass Architecture. *Transactions of the Association for Computational Linguistics*, 2:273–284.

H. Daumé III. 2004. Notes on CG and LM-BFGS Optimization of Logistic Regression. Paper available at `http://pub.hal3.name`, implementation available at `http://hal3.name/megam/`, August.

P. Denis and P. Muller. 2010. Comparison of different algebras for inducing the temporal structure of texts. In *Proceedings of Coling 2010*, pages 250–258, Beijing.

L. Derczynski and R. Gaizauskas. 2013a. Temporal Signals Help Label Temporal Relations. In *Proceedings of the annual meeting of the Association for Computational Linguistics (ACL)*, pages 645–650. ACL.

L. Derczynski and R. Gaizauskas. 2013b. Empirical Validation of Reichenbach's Tense Framework. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS)*, pages 71–82.

L. Derczynski, H. Llorens, and N. UzZaman. 2013. TimeML-strict: clarifying temporal annotation. *arXiv preprint arXiv:1304.7289*.

L. Derczynski. 2017. *Automatically Ordering Events and Times in Text*. Springer.

Q. X. Do, W. Lu, and D. Roth. 2012. Joint inference for event timeline construction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 677–687. ACL.

C. Freksa. 1992. Temporal reasoning based on semi-intervals. *Artificial Intelligence*, 54(1):199–227.

R. Gaizauskas, E. Barker, C.-L. Chang, L. Derczynski, M. Phiri, and C. Peng. 2012. Applying ISO-Space to Healthcare Facility Design Evaluation Reports. In *Seventh Workshop on Interoperable Semantic Annotation (ISA), Eighth International Conference on Language Resources and Evaluation*, pages 13–20.

G. Glavaš and J. Šnajder. 2013. Recognizing Identical Events with Graph Kernels. In *Proceedings of the annual meeting of the Association for Computational Linguistics (ACL)*, pages 797–803. ACL.

J. Hobbs and J. Pustejovsky. 2003. Annotating and reasoning about time and events. In *Proceedings of AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*, SS-03-05, pages 1–9.

D. Hovy, J. Fan, A. Gliozzo, S. Patwardhan, and C. Welty. 2012. When Did that Happen? - Linking Events and Relations to Timestamps. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 185–193. ACL.

R. Lévi. 1927. Théorie de l'action universelle et discontinue. *J. Phys. Radium*, 8(4):182–198.

I. Mani, B. Wellner, M. Verhagen, and J. Pustejovsky. 2007. Three approaches to learning TLINKS in TimeML. Technical Report CS-07-268, Brandeis University, Waltham, MA, USA.

T. Miller, S. Bethard, D. Dligach, S. Pradhan, C. Lin, and G. Savova. 2013. Discovering Temporal Narrative Containers in Clinical Text. In *Proceedings of the Workshop on Biomedical Natural Language Processing*, pages 18–26. ACL.

S. Mirroshandel, G. Ghassem-Sani, and A. Nasr. 2011. Active Learning Strategies for Support Vector Machines, Application to Temporal Relation Classification. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, pages 56–64.

J. Pustejovsky and A. Stubbs. 2011. Increasing informativeness in temporal annotation. In *Proceedings of the Linguistic Annotation Workshop*, pages 152–160. ACL.

J. Pustejovsky, P. Hanks, R. Sauri, A. See, R. Gaizauskas, A. Setzer, D. Radev, B. Sundheim, D. Day, L. Ferro, et al. 2003. The Timebank Corpus. In *Proceedings of the Corpus Linguistics Conference*, pages 647–656.

J. Pustejovsky, B. Ingria, R. Sauri, J. Castano, J. Littman, and R. Gaizauskas. 2004. The Specification Language TimeML. In *The Language of Time: A Reader*, pages 545–557. Oxford University Press.

J. Pustejovsky, K. Lee, H. Bunt, and L. Romary. 2010. ISO-TimeML: An International Standard for Semantic Annotation. In *Proceedings of the International conference on Language Resources and Evaluation (LREC)*, pages 394–397. European Language Resources Association.

J. R. Quinlan. 1986. Induction of decision trees. *Machine learning*, 1(1):81–106.

M. Sabou, K. Bontcheva, L. Derczynski, and A. Scharl. 2014. Corpus Annotation through Crowdsourcing: Towards Best Practice Guidelines. In *Proceedings of the Conference on Language Resources and Evaluation (LREC)*, pages 859–866. ELRA.

W. Schaeken and P. N. Johnson-Laird. 2000. Strategies in temporal reasoning. *Thinking & Reasoning*, 6(3):193–219.

A. Scheuermann, E. Motta, P. Mulholland, A. Gangemi, and V. Presutti. 2013. An empirical perspective on representing time. In *Proceedings of the international conference on Knowledge Capture*, pages 89–96. ACM.

A. Setzer and R. Gaizauskas. 2000. Annotating events and temporal information in newswire texts. In *Proceedings of the Second International Conference On Language Resources And Evaluation (LREC)*, 321, pages 1–7. European Language Resources Association.

J. Strötgen and M. Gertz. 2016. *Domain-Sensitive Temporal Tagging*. Morgan Claypool.

N. UzZaman, H. Llorens, L. Derczynski, M. Verhagen, J. F. Allen, and J. Pustejovsky. 2013. SemEval-2013 Task 1: TempEval-3: Evaluating Time Expressions, Events, and Temporal Relations. In *Proceedings of SemEval*, pages 1–9. ACL.

M. Verhagen, R. Gaizauskas, F. Schilder, M. Hepple, J. Moszkowicz, and J. Pustejovsky. 2009. The TempEval challenge: identifying temporal relations in text. *Language Resources and Evaluation*, 43(2):161–179.

M. Verhagen, R. Saurí, T. Caselli, and J. Pustejovsky. 2010. SemEval-2010 task 13: TempEval-2. In *Proceedings of SemEval*, pages 57–62. ACL.

M. Verhagen. 2004. *Times Between The Lines*. Ph.D. thesis, Brandeis University.

M. B. Vilain and H. A. Kautz. 1986. Constraint Propagation Algorithms for Temporal Reasoning. In *Proceedings of the AAAI Conference*, volume 86, pages 377–382.

K. Yoshikawa, S. Riedel, M. Asahara, and Y. Matsumoto. 2009. Jointly identifying temporal relations with Markov logic. In *Proceedings of the annual meeting of the Association for Computational Linguistics (ACL)*, pages 405–413. ACL.