Disambiguating Temporal Connectives into TimeML relations

B006745
Master of Science
Speech and Language Processing
The University of Edinburgh

2011

Contents

List of Tables

List of Figures

Abstract

The project is about learning temporal relations from unannotated text. This effort builds on the work of Lapata and Lascarides (2006), who developed a system that uses temporal connectors (*after, before, while, when, as, once, until* and *since*) in unannotated text to build a system to determine intra-sentential temporal relations.
In an extension of this approach, they used their system to determine TIMEML relations (BEFORE, INCLUDES, BEGINS, ENDS and SIMULTANEOUS) between events. Since temporal connectors do not translate one-to-one to TIMEML relations, the main focus of this project is on disambiguating the temporal connectors into TIMEML relations to preprocess the training data and use the system to directly learn the TIMEML relations. This will be done using a rule-based system and will be evaluated on the TIMEBANK corpus.

1 Introduction

Temporal Semantics plays a pivotal role in many language processing tasks, such as information retrieval, question answering, recognising textual entailment and text summarisation. The identification of events and the understanding of how they relate to each other is crucial in any text understanding task. For example, in a question answering system, questions like the following are beyond the scope of any QA system, if it lacks temporal semantics (Pustejovsky et al. 2003a: 2).

(1) a. Is Gates currently the CEO of Microsoft?
b. When did Iraq finally pull out of Kuwait during the war in the 1990s?
c. Did the Enron merger with Dynegy take place?

Historically, approaches to the subject of temporal semantics in computational linguistics have been mostly inference-based, rooted in symbolic AI (cf. Lapata and Lascarides 2006; Mani et al. 2005). But, as Lapata and Lascarides (2006: 88) point out, theory based systems inevitably lack coverage, because they are built on simplified assumptions about language. Because of this, more recent work has followed other fields in computational linguistics and has strived to "place more emphasis on the study of real language as evidenced in collections of texts [...]. Linguistic judgements about these texts are then recorded in the forms annotations associated with the texts [...]." (Mani et al. 2005: 487)
A corpus-based approach to the temporal evaluation task can be broken down into the following three tasks:

Task 1: Annotating times

Task 2: Annotating events

Task 3: Annotating temporal relations

- between times and events
- between events
  - inter-sentential
  - intra-sentential

The first task deals with purely temporal entities, namely temporal expressions that might refer to "times (*five to eight*), dates (*July 1, 1867*) durations (*three months*), or frequencies or sets of regularly recurring times (*weekly*)" (Verhagen et al. 2009: 162). The second task is to annotate the linguistic expressions that denote events, which usually are verbs, the predicates of a sentence, but not necessarily (Verhagen et al. 2009: 163). The third task is to annotate temporal relations between those events (event-event relations), as well as anchoring events in time (time-event relations). Event-event relations may be inter-sentential or intra-sentential.

This last task of annotating event-event relations is not only difficult from a computational point of view, but "is sufficiently complex that human annotators can realistically identify only a small number of the temporal relations in text, thus compromising recall." (Lapata and Lascarides 2006: 86)

The difficulty lies not only in indentifying, where temporal relations apply, but also in choosing which relation applies: Setzer and Gaizauskas (2001) did a study with human annotators, reporting an average of 40/68 precision/recall for annotating temporal relations (the possibility of this task is bounded by the correct annotation of events, so a low performance in Task 2 lowers recall for Task 3). Verhagen et al. (2009) report similar figures regarding the preparation of the corpus for the TempEval challenge, even though they used only three temporal relations. (see Sections 2 and 3.1). Setzer and Gaizauskas (2001) identify five possible reasons for this low figure, three of which boil down to "human error" (imprecision of the guidelines, lack of training/experience, annotator fatigue, annotator carelessness) and only one to the task itself, although it is easily the most difficult to overcome:

> Consider the sentence *All 75 people on board the Aeroflot Airbus died when it ploughed into a Siberian mountain in March 1994.* Is the relation between the passengers dying and the plane crash one of causality and given that, did the passengers die, *after* the plane crash? Or is a plane crash an event that contains many subevents and is the death of the passengers part of it, which would imply that the death occurred *during* the plane crash? Or did they happen roughly at the same time, a relation covered by our temporal relation *simultaneous*? (Setzer and Gaizauskas 2001: 8)

The inherent difficulty of temporal relation annotation only increases with the introduction of more relations. TimeML (Section 3.1 and Allen's Interval Algebra (Allen 1983), on which most temporal relation marking schemas are based, allow for very fine-grained differences, adding the possibility that the passengers died IAFTER[1] – immediately after – the plane crashed, or that the plane crash started at the same time as the dying of passengers, in which case BEGINS would be the appropriate marking. The bigger the inventory, the higher the imprecision.

Or, in the words of Steedman (2010):

[1] Incidentially, this is the example sentence to illustrate the meaning of IBEFORE in the TimeML annotation guidelines. (Pustejovsky et al. 2002: 45)

> The general suspicion is that this is because such hand-built resources are both too high-level in terms of ontology, and too small-scale in comparison to what a mixture of animal evolution and social learning has put in our heads.

The mathematical beauty of a complete computational system like Allen's interval algebra is thus compromised by the impreciseness of human cognition. The only choices seem to be either to stick with it, or to introduce 'fuzzy' relations which closer resemble our intuitions about event ordering and give up on the inferential power of the calculus (cf. Setzer et al. 2005: 583). Somewhat ironically, Allen's interval algebra was designed with imprecision as a governing principle, albeit in a different understanding: "The representation should allow significant imprecision. Much temporal knowledge is strictly relative [...] and has no relation to absolute dates." (Allen 1983: 833).

In summary, temporal evaluation is not only crucial for various NLP tasks, but also overall difficult, for humans as well as machines, and therefore one of the most interesting tasks in computational linguistics today.

This project is concerned with the annotation of intra-sentential event-event relations. It is an extension of the system of Lapata and Lascarides (2006), who used overt temporal connectors, *after, as, before, once, since, until, when* and *while*, to train a system that could be used to insert them into text where they are not present, thus also being able to predict temporal relations. They also trained their classifier on purely temporal relations. Because mapping from temporal relations as defined in TimeML to temporal connectors is not one-to-one, they randomly split the ambigous temporal connectors among the temporal relations that are applicable to them, and evaluated the classifier on TimeBank with promising results.

Their approach is one of few, if not the only one so far, that tried to train a classifier for temporal relations on unlabeled data. Unsupervised learning in this area can really only deal with temporal relations through proxy, and temporal connectors seem a promising choice for circumvention of the problem of data sparsity that is especially prevalent in this area of computational linguistics, and to take advantage of learning from huge corpora that has already benefitted a wide variety of fields within NLP today, by using an indirect label of temporal relationship and turning it into an explicit one, incorporating linguistically informed features.

The idea is to provide a reliable mapping of temporal connetors to temporal relations by disambiguating the temporal connectors into TimeML relations before training.

The disambiguator will be rule-based, for reasons explained in Section 3.1. The most promising features to base rules on seem to be tense and grammatical aspect (discussed in Section 3.3) and lexical aspect (discussed in Section 3.4) based on the theoretical foundations layed out in Dorr and Gaasterland (2007) and Dorr and Olsen (1997).

Evaluation will be performed by a) using only the disambiguator on TimeBank, to test it on its own and see if the defined rule set is appropriate for the task, and most importantly, the disambiguator will be used on the Bllip corpus, the same corpus the original Lapata and Lascarides (2006) classifier was trained on, to see if disambiguating the training data into TimeML relations improves performance.

In the following sections, I will first explore related work, then I will explain the tools and concepts this project is based on in relevant detail in Section 3. Section 4 details the workings of the disambiguator, which will be evaluated in Section 5. A discussion of the performance of the system follows in Section 6.

2 Related Work

A wider interest in corpus-based approaches to temporal semantics is a rather recent phenomenon; Mani et al. (2005: 488) date it to the late 1990s. The TimeML annotation language was first released in 2003, and a dedicated TempEval challenge that evolved from SemEval was first held in 2007 (Verhagen et al. 2009).
Earlier attempts at temporal evaluation involve common sense ontologies like CYC (to which the quote from Steedman (2010) in Section 1 relates), and projects in its vein, who are trying to exploit the notion that temporal information is transferred not only by grammatical means but also implicitly derived through world knowledge: It is a well-established fact that only the salient parts are expressed when communicating. One specialised example of such an ontology is EventNet, a LifeNet derived ontology that encodes the temporal relations between commonsense events (Espinosa and Lieberman 2005) and makes it accessible for computational purposes.
An example for an elaborate symbolic system is Lascarides and Asher (1993), who built an extensive model based on defeasible logic that involves non-monotonic reasoning and requires extensive domain knowledge. The advantage of a complex system like this is that it functions very well as a general model of human cognition, but in a use-case context, the amount of world knowledge and the reasoning processes required are too vast to be feasible. Furthermore, as Lapata and Lascarides (2006: 88) point out, any system involving non-monotonic reasoning becomes intractable, especially when facing large amounts of data.
But even though symbolic models are not efficient, the theoretical work expressed in them can help find expressive features for corpus-based approaches (cf. Lapata and Lascarides 2006: 110).
Accordinlgy, the goal in this project is to keep it simple, and to try finding an efficient minimal set of rules which are rooted in actual real world data.
The availability of human-annotated corpora allows for supervised machine-learning approaches: One of the first attempts at automated learning of temporal relations is Mani et al. (2006), who trained a Maximum Entropy classifier on TimeBank, using transitive closure of temporal relations (i.e. if A before B and B before C then A before C) to successfully increase the amount of training instances, a principle that is elaborated upon in Chambers and Jurafsky (2008).
The features used were only the ones available in the TimeBank event tag. Their classifier outperformed all of the rule-based systems they compared it with (including the ontology based ones), although one should not forget that this system was sort of overfitted to this particular domain, trained on perfect features. Chambers et al. (2007)

built a system based on the Mani et al. (2006) system, but with additional features, most of which were taken from Lapata and Lascarides (2006). They could show a slightly better performance with the enhanced feature set over Mani et al. (2006) as well as Lapata and Lascarides (2006).
The TempEval challenge saw a variety of different approaches to the challenge of annotating times, events and temporal relations, one of them a shallow classifier trained only on the provided data-sets without using any deep NLP analysis that could be viewed as a baseline (Verhagen et al. 2009: 170). Most other systems used a combination of machine learning and rule-based approaches, and with only the one exception, they all used syntactically and semantically informed features. Alas, most systems did not differ significantly from the baseline provided by the shallow classifier: This result in conjunction with the results by Mani et al. (2006) and Chambers and Jurafsky (2009) seem to suggest that there is a ceiling effect regarding the performance of temporal relation annotiation systems. One system developed by Puşcaşu (2007) stood out, however, and it used mainly knowledge-based and statistical methods. Interestingly, their system for annotating intra-sentential event-event relations is mainly based on tense, connector (temporal or otherwise) and dependency relations between clauses (cf. Puşcaşu 2007: 486). This gives valid reason to be confident in the overall possibility of the disambguation task at hand here, with the tools chosen, even if the TempEval challenge used a different set of temporal relations (see Section 3.1).
The problem with supervised automated approaches in this area of computational semantics is data sparsity: The only available corpus annotated with temporal relations is TimeBank, which is rather small (see Section 3.1 for statistics), even when expanded through transitive closure. Another approach to overcome the problem of data sparsity is to use unsupervised learning: As already mentioned, this necessitates the need for a proxy that can be used to learn temporal relation indirectly. Chambers and Jurafsky (2009) is an approach to learn 'narrative schemas', i.e. the typical participants and their actions in a specific situation. This could possibly be used to infer temporal relations based on the typical ordering of tasks, like it is encoded in ontologies like EventNet. This is an interesting method, in that it brings the concept of world knowledge back into the picture in a computationally managable way.
The other main source of this project are Dorr and Gaasterland (2007), whose system will be described in great detail in Sections 3.3 and 3.4. Suffice to say here, that they employ a rule-based system, that was developed in the conetxt of machine translation to accurately translate sentences via an interlingua that is oriented at a lexical conceptual structure (cf. Dorr 1992b). This principle was then employed in Dorr and Olsen (1997) for other NLP tasks as well, and in its recent form is implemented in a multi-document summarisation system (cf. Dorr and Gaasterland 2007). The long development cycle of this project suggests that it is well thought out, as well as easily deployable. The most thorough account of the idea behind all these projects can be found in Dorr and Gaasterland (2002).
As far as concerns the related work in this particular area, and although the work by

Dorr provides a great foundation from which to start, nothing like this project has been entertained yet, and the hope is that it can open a door for unsupervised methods in the area of temporal relation annotation.

3 Background

As mentioned before, this project is primarily based on Lapata and Lascarides (2006) and Dorr and Gaasterland (2007), Dorr and Olsen (1997). Both of these systems were developed from a summarisation inspired point of view, with Lapata and Lascarides (2006) trying to insert temporal connectors into generated text and Dorr and Gaasterland (2007) describing part of a sentence-ordering module in a multi-document summarisation system. This project is not designed for a specific use-case, but rather in the spirit of the TempEval challenge, trying to serve as a tool for temporal relation annotation between events.

When conceiving this project, the original plan was to train the disambiguator on TimeBank, but this proofed infeasible due to the small size of TimeBank, containing only a small number of temporal connectors (see Section 3.1 for statistics on this). Another TimeML annotated corpus is the AQUAINT TimeML corpus (also called Opinion Corpus), but it is only half the size of TimeBank, so even when putting the two corpora together, the number of instances of temporal connectors is still too small to train a robust disambiguation system. In addition to the small number of connectors in TimeBank and the opinion corpus, these two are also the only TLink annotated corpora available for testing, which means that the number of available training data would be further reduced, since the data would have to be split into a test and a training set. In that case, the test data would probably be too small to deliver meaningful results. There is also the WikiWars Corpus (Mazur and Dale 2010), which is about double the size of TimeBank, but it is annotated in Timex2 only, i.e. it contains only annotation of temporal expressions and events (Task 1 and 2), but not of temporal relations between events and thus cannot be used in this task, neither for evaluation nor training.

In light of this, the disambiguator would have to be rule-based. The system described in Dorr and Gaasterland (2007) lays the foundation for extracting intra-sentential temporal relations using tense and aspect (both grammatical and lexical).

The shortcomings of rule-based approaches has already been touched upon in the Introduction. To avoid falling into that trap, the rules were acquired from in-domain real life data, namely the Penn Treebank Wall Street Journal Corpus. This is the same corpus that Dorr and Gaasterland (2007) used to verify their theoretical assumption, so their approach is also grounded in real life data and not too abstract to be widely applicable, even though it is mainly based on observations on toy examples, (i.e. by combining example sentences comprised of only a subject and a verb with a certain lexical aspect that is cycled through all tenses and aspects with another such sentence and listing all the possible temporal relations that can theoretically hold between these sentences, so their approach is not enough to disambiguate the data, but only limits the number of

choices that can be made under certain (grammatical) conditions.
In this project, I make the assumption that in real world data, especially in the domain of news articles, i.e. a text type whose main purpose is to relate information about events, an actual temporal relation will be specified and extractable. Starting from this assumption, it should be possible to detect patterns in the data and make them into rules.
Combining the rule-based approach of Dorr and Gaasterland (2007) with the approach of Lapata and Lascarides (2006) might give us the advantage of benefitting from both a rule-based system and a machine-learning system. The following sections describes the foundations on which the disambiguator is built.

3.1 Temporal Relations

The de-facto annotation standard for temporal relations is TimeML[2], and its accompanying corpus is TimeBank[3] which both were released in 2003. The latest version of TimeBank is 1.2.1, released in 2006. TimeML is an extension of the Timex2 markup language, an XML annotation standard for dates and times. Timex2 enables annotations of dates and times (Task 1), TimeML adds the capability of tagging events (Task 2) and relations between these events, as well as between times and events (Task 3).
EVENTs are anything that can *happen* or *occur*, usually a verb, but not necessarily (Pustejovsky et al. 2003a: 3). The similarity between events in the TimeML sense and verbs in general is mirrored in the fact that the EVENT tag in TimeML categorises events into classes much like verb classes and also records the tense and grammatical aspect of the expression, as well as modality and polarity, as attributes of the EVENT tag (Pustejovsky et al. 2003a: 4).
Furthermore, TimeML provides three different LINK tags that specifiy different kinds of relations between events, or between times and events.

1. **TLINK**: represents the temporal event-event relation or event-time relation.
2. **SLINK**: non-temporal relations between events (e.g. modal, referring to other possible worlds).
3. **ALINK**: 'aspectual' relations between events, i.e verbs that modify other events, namely initiates, culminates, terminates or continues an event.

With the additional SLINK, TimeML captures the fact mentioned in Section 2, that events may be related in non-temporal ways, even though that relation is expressed with temporal means like modality, which is a part of the TAM (tense-aspect-modality) complex. Such relations may be

1. Modal: relating to hypothetical events.
2. Factive: an event relating to the veracity of another event

[2] `http://timeml.org/site/index.html`, see also Pustejovsky et al. (2003a) and Pustejovsky et al. (2002)
[3] `http://timeml.org/site/timebank/timebank.html`, see also Pustejovsky et al. (2003b)

3. Counterfactive: an event relating to the non-veracity of another event
4. Evidential: events that are reported or perceived
5. Negative Evidential: events that are reported or perceived as not having taken place
6. Negative: negated events

Temporality and causation are intricately interwoven in our language. The proximity of reported events is taken as a clue for, not necessarily a causative realtionship, but what Moens and Steedman (1988) call a 'contingency' relationship: they explain the status of the *when*-connector as not-strictly-temporal, but also not-strictly-causal, with the concept of contigency, i.e. the expectation of the reader is that the events combined by the connector are involved in a relationship that is beyond pure temporality. They give the example of a chain of causative relationships, in which only the next event can be inferred, i.e. the relation is not transitive. The splitting of the during relation also captures a relation beyond pure temporality, since it assumes that one event starts (or ends) the other, i.e. some kind of contigent relationship between the events is assumed. Consequently, causation is also dealt with in TimeML, albeit not in its own tag, but rather as a possible inference given certain combinations.
TimeML also features a Signal tag, that is used to mark linguistics expressions that give clues about temporal relationships, such can be temporal prepositions like *during*, or temporal quantifiers such as *twice*, and also temporal connectors of the kind discussed in the next section (Pustejovsky et al. 2003a). The Signal tag will be relevant for the first evaluation in Section 5.1.

Corpus → ↓ *No. of ...*	**TimeBank**	**Opinion Corpus**	**PT3** WSJ	**Bllip**
articles	185	73	2,499	98,732
words	61K	35K	1M	30M
TLinks	6418	5365	n\a	n\a
temporal connectors	181		2686	90862

Table 1: Corpora Statistics

The TimeML relations, specified as the `relType` attribute in TLinks, are based on Allen's Interval Algebra (Allen 1983), pictured in Table 2. There are, however, some differences between TimeML relations and Allen relations: Allen's relations allow for multiple relations to be valid between two events, while TimeML forces annotators to pick only one. The original TimeML specification also doesn't include overlap (Pustejovsky et al. 2003a), although the TempEval challenge was based around the simplified tag-set of before, after and overlap, with additional tags for ambiguous cases before-or-overlap, overlap-or-after and vague (cf. Verhagen et al. 2009: 167). This overlap tag, however, captures all events where the time-line of the two events isn't strictly distinct (as it is in the before and after relations), whereas Allen's overlap (o, oi) relation is distinct from the during relation and does not hold

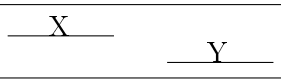
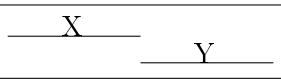
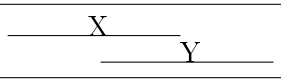
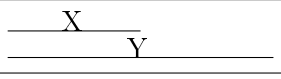
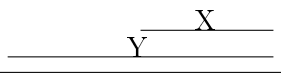
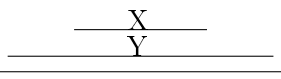
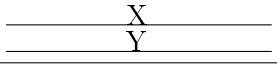
Allen	TLink	Illustration
X < Y Y > X	BEFORE AFTER	
X m Y Y mi X	IBEFORE IAFTER	
X o Y Y oi X	OVERLAP	
X s Y Y si X	BEGINS BEGUN_BY	
X f Y Y fi X	ENDS ENDED_BY	
X d Y Y di X	IS_INCLUDED INCLUDES	
X = Y	SIMULTANEOUS	

Table 2: Allen Relations - TimeML relations

where one event is contained in the other.

The whole set of `relTypes` is BEFORE, AFTER, INCLUDES, IS_INCLUDED, DURING, DURING_INV, SIMULTANEOUS, IAFTER, IBEFORE, IDENTITY, BEGINS, ENDS, BEGUN_BY, ENDED_BY. The difference between IDENTITY and SIMULTANEOUS is that the former relates identical events, the latter events that occupy the same space in time. The difference between DURING and IS_INCLUDED is that DURING applies especially to events, but mostly times, that express a duration (Pustejovsky et al. 2002: 45).

It can be said that in general, every annotation task uses its own set of relations as they see fit, for the reasons discussed in the introduction. The most common reduction is to fold inverse relations into one relation. As has already been mentioned the relation of events in real text is not always as clear-cut as the strict definitions in Allen's algebra (Setzer et al. 2005: 583), and the decision, which temporal relations are useful, is dependant on the task.

A markup for a sentence containing a temporal connector extracted from the TimeBank corpus is shown in Figure 1.

3.2 Temporal Connectors

The editors of Mani et al. (2005) state in their reader on temporal semantics in computational linguistics, that event-event relations may be either implicit or explicit and that "[t]he primary mechanism for explicit relation is the temporal conjunction, typically used to relate the event expressed in a subordinated clause to one in a main clause" (Mani et al. 2005: 496). This is also the main idea behind the system developed by Lapata and Lascarides (2006): to train a model on explicit temporal connectors that is able to insert these connectors where they are not present, e.g into a sentence like (2),

```
He <EVENT eid="e68" class="REPORTING">said</EVENT>
     <EVENT eid="e69" class="OCCURRENCE">construction</EVENT> would n't
     <EVENT eid="e70" class="ASPECTUAL">resume</EVENT>
     <SIGNAL sid="s227">until</SIGNAL> market conditions
     <EVENT eid="e73" class="OCCURRENCE">warrant</EVENT> it.

<MAKEINSTANCE eventID="e68" eiid="ei2051"
      tense="PAST" aspect="NONE" polarity="POS" pos="VERB"/>
<MAKEINSTANCE eventID="e69" eiid="ei2052"
      tense="NONE" aspect="NONE" polarity="POS" pos="NOUN"/>
<MAKEINSTANCE eventID="e70" eiid="ei2053"
      tense="NONE" aspect="NONE" polarity="NEG" pos="VERB" modality="would"/>
<MAKEINSTANCE eventID="e73" eiid="ei2054"
      tense="PRESENT" aspect="NONE" polarity="POS" pos="VERB"/>

<TLINK lid="143" relType="AFTER"
      eventInstanceID="ei2053" relatedToEventInstance="ei2054" signalID="s227"/>
<SLINK lid="l132" relType="MODAL"
      eventInstanceID="ei2054" subordinatedEventInstance="ei2053"/>
<SLINK lid="l103" relType="EVIDENTIAL"
      eventInstanceID="ei2051" subordinatedEventInstance="ei2053"/>
<ALINK lid="l127" relType="REINITIATES"
      eventInstanceID="ei2053" relatedToEventInstance="ei2052"/>
```

Figure 1: Example TimeML markup for a sentence with a temporal connector: *He said construction wouldn't resume until market conditions warrant it.*

thus also providing a model capable of inferring that John kissed the girl after he had met her.

(2) John kissed the girl he met at a party.

To do this, they trained a classifier on the overt temporal connectors *after, as, before, once, since, until, when* and *while*, testing different conjunctive and disjunctive combinations of the following featureset:

1. **Temporal Signature (T)**
   Tense and grammatical aspect of the verb.
2. **Verb Identity (V)**
   Exploiting the lexical relationship between supersenses of verbs.
3. **Verb Class ($\mathbf{V_L}$, $\mathbf{V_W}$)**
   Levin verb classes and WordNet supersenses.
4. **Noun Identity (N)**
   Information about the relationship between subjects (e.g. part-whole), based on WordNet supersenses.
5. **Noun Class ($\mathbf{N_W}$)**
   WordNet supersenses.

Relation	Connector
BEFORE	after, before, once, when after, before, since, until, when
INCLUDES	as, when, while when, while
SIMULTANEOUS	as, when, while when, while
BEGINS	since until, when, while
ENDS	until after, before, since, when, while

Table 3: Mapping of temporal connectors to temporal relations. The upper line in each cell shows the mapping used by Lapata and Lascarides (2006) and the lower line the mapping by Dorr and Gaasterland (2007)

6. **Adjective (A)**
   Temporal Adjectives can provide clues about the temporal ordering.
7. **Syntactic Signature (S)**
   Number of NPs, VPs, PPs, ADJPs and ADVPs contained in the parse tree.
8. **Argument Signature (A)**
   Captures direct and indirect objects of the verb and whether it is modified by an adverbial or preposition.
9. **Position (P)**
   If the main clause is sentence initial or not.

Conjunctive model in this context means that the features of main and subordinate clause are assumed to be interdependent, whereas in the disjunctive model, features are assumed to be independent. In the first experiment, they tested their models on a training set from the Bllip corpus that contained sentences with their connector removed. Their results found that a disjunctive model using only Verb Identity and Syntactic Signature outperformed all other combinations of features.
In the next step, they used an ensemble training method combining several classifier models that use different feature combinations. These classifiers are trained individually and their output is combined in a decision tree (Lapata and Lascarides 2006: 102-104). They again trained a conjunctive and a dsjunctive version, both of which outperformed the individual models. The disjunctive model again outperformed the conjunctive model. The feature combinations of the submodels of the disjunctive classifier is pictured in Table 4.
The most successful feature was syntactic signature, and it is consequently a feature in every submodel. Other important features are verb identity and position, which are almost in every submodel.

Disjunctive Ensemble Submodels						
AN_WNPSV	APSV	ASV	$PRSV_W$	PSV_N	SV_L	NPRSTV
PRS	PRST	PRSV	PSV	SV		

Table 4: Disjunctive Ensemble Submodels:
A: adjectives, V: verb identity, V_W: WordNet verb supersenses, V_L: Levin verb classes, N: noun identity, N_W: WordNet noun supersenses, P: clause position, R: argument signature, S: syntactic signature, T: tense signature.

In another experiment, they trained a classifier for TimeML relations using the disjunctive ensemble method. Like the quote in the beginning of this chapter suggests, temporal connectors and TimeML relations "are more or less semantically compatible" (Lapata and Lascarides 2006: 106) and so a mapping from temporal connectors to TimeML relations is possible. However, this mapping is not one-to-one, since one connector can be used to express several TimeML relations. Lapata and Lascarides (2006) randomly split the ambiguous connectors among the corresponding TimeML relations, adding a number randomly put together sentences for training a NO-TEMP-REL for cases where no temporal relation holds between events. They admit that this split is "far from perfect" (Lapata and Lascarides 2006: 108), mainly because the ambigous connector since had to be assigned to the BEGINS relation exclusively for lack of another representative connector.
Furthermore, they reduced the set of TimeML relations by collapsing inverse relations (INCLUDES – IS_INCLUDED, BEFORE – AFTER, BEGINS – BEGUN_BY and ENDS – ENDED_BY) and collapsing IBEFORE and IAFTER into BEFORE. The resulting set of eight relations and the mapping of the connectors to TimeML relations is pictured in Table 3. The table also contains the mappings of Dorr and Gaasterland (2007), and contains a conflict regarding the connectors *since* and *until*, a short discussion of this follows in Section 4.1.

	TimeBank		Bllip	
	No.	%	No.	%
after	56	30.9	13,228	15.9
as	14	7.7	15,904	19.0
before	23	12.7	6,572	7.8
once	5	2.8	638	0.8
since	17	9.4	2,742	3.3
until	25	13.8	5,307	6.3
when	35	19.2	35,895	42.8
while	6	3.3	3,524	4.2
Total	181	100	83,810	100

Table 5: Number of instances of temporal connectors in TimeBank and Bllip (TimeBank Statistics according to `http://timeml.org/site/timebank/browser_1.2/displayTags.php?tagtype=signal`, Bllip statistics according to (Lapata and Lascarides 2006: 93))

3.3 Tense and Grammatical Aspect

The tense of a verb can be one of *past, present* or *future*. To explain how tense relates the time of the utterance to the event uttered, we can assume a speech time S and an event time E, such that S and E are identical for the present tense, S is after E for the past tense, and E is after S for the future tense. This gives us an insight into how events can be ordered by their tense if we have a reference time, in the case of a news article, this would be the document creation time, in the case of an utterance, it would be the time of the utterance.

Grammatical aspect in English can be one of *perfect, progressive* or *unmarked*. The grammatical aspect is a non-inherent feature of verbs, i.e. it can be derived from its surface from. The English aspectual system differs from that in languages with a dedicated aspectual system in several ways: Firstly, the grammatical aspect is closely associated with the tense (hence the names "past perfect" and so on). Secondly, the English language offers the option to leave the grammatical aspect unmarked by using a simple tense. The English tense-aspect combination can be described with the simple tense as being unmarked, the progressive describing a process, and the perfect describing a culmination, (cf. Moens and Steedman 1988: 18-22). The term ' aspect' describes the point of view that the speaker has on the event: a progressive highlights the ongoing event, i.e. the event in its progression, whereas the perfect moves the focus to the achieved event, mostly highlighting the result of the event in question.

English perfect really serves a double purpose, on the one hand it marks an event as finished and draws focus to its result, a function that is associated solely with grammatical aspect in other languages, on the other hand it simply marks anteriority of an event relative to another time: To explain the difference between past simple and past perfect, we need to introduce a third point in time, a reference point R that refers to a point between an event and the time of speech. In the simple tenses, the reference point is identical with the event time. In perfect tenses, the reference point lies after the event time. Progressive tenses can then be seen as intervals rather than points in this schema (cf. Reichenbach 2005: 71-73).

It seems intuitive to assume that most of the temporal information about events is encoded in the tense and aspect of the verb expressing said event. This assumption is also made in TimeML, where the event tags contain this information about events.

However, linguistic means to express temporality, apart from unexpressed temporal relations that need to be resolved with world knowledge, do not only express temporal relations. The problem is twofold: A temporal expression may also express other relations, such as causal relations, or a temporal expression may be ambiguous and convey several temporal relations. The system of tense, aspect and modality is multifunctional in that respect, and it probably is mostly a reflection of how we process time, events and causality. Due to the way humans process and store information, temporal information is not neatly separable from information about how events relate to each other in ways other than purely temporally.

	progressive	perfect	simple	gerund	total
progressive	0.13	0.05	2.29	0.01	2.48
perfect	0.05	0.49	4.45	0.03	5.02
simple	1.88	1.99	84.38	0.77	89.02
gerund	0.09	0.15	3.17	0.07	3.48
total	2.15	2.69	94.29	0.87	

Table 6: Percentage of aspect combinations in corpus

	present	past	future	gerund	total
present	27.35	9.30	0.30	0.54	37.48
past	8.79	45.02	0.06	0.24	54.11
future	4.67	0.16	0.08	0.02	4.93
gerund	1.95	1.44	0.02	0.07	3.48
total	42.75	55.92	0.45	0.87	

Table 7: Percentage of tense combinations in corpus

If we consider example (2) again, in a sentence like 3, the temporal relation is marked explicitly by using the present perfect tense: The event marked with a perfect aspect is therefore finished before the other event takes place.

(3) John kissed the girl he met at a party.

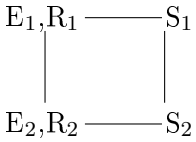
(4) John kissed the girl after he had met her at a party.

A reason for not explicitly marking the temporal relation, instead relying on the listener to know the ordering in which such events usually occur becomes clearer when we consider sentence 4: a version that would express all the details relevant to the temporal ordering of events would draw the attention of the reader away from the events expressed: the overspecification of the temporal details implies to the reader that the temporal ordering is somehow of significance. The reason for this is expressed the Gricean maxime "Be relevant": if a temporal ordering is assumed to be clear to the reader, be it through world knowledge, i.e. that you first have to meet somebody before you can kiss them, or other means, no effort is made to make it more explicit by using a marked tense, as was explained through Example 4.
Another way to mark temporal relations explicitly through tense is to mix tenses as in(5):

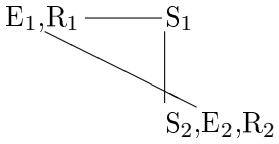
(5) Trustcorp Inc. will become Society Bank & Trust when its merger is completed with Society Corp. of Cleveleand, the bank said.

Dorr and Gaasterland (2007) describe a constraint on the mixing of tenses: They consider a Reichenbachian tense structure called Complex Tense Structure (CTS) and a formulate constraints on this structure, called Constraint on Derived Tense Structure

(CDTS): In a pair of main clause and subordinate clause, the three time points speech time S, event time E and reference point R are associated to each other like so (for past simple - past simple):



The constraint states, that these associations between the time points of a CTS are not allowed to crossover. In the case of a present simple - past simple combination, this constraint is violated:



This takes also the grammatical aspect of the perfect into account, since none of the association from any of the time points are allowed to crossover. The list of allowable tense combinations is pictured in Table 8.

Main clause (matrix)	**Subordinate Clause (adjunct)**
{fut, fut perf}	{pres, pres perf, fut, fut perf}
{past, past perf}	{past, past perf}
{pres, pres perf}	{pres, pres perf}

Table 8: Allowable tense combinations according to CDTS

So, tense and aspect can encode a temporal ordering. But, as is visible from Tables 7 and 6, unmarked tenses are by far the most common, accounting for over 80% of total combinations, and within-tense combinations make up a total of over 70%. Obviously, additional features will need to be considered for disambiguation.

3.4 Lexical Aspect and Intervals

The lexical aspect, also known as *aktionsart* or *aspectual type*, is an inherent feature of the verb and as such cannot be derived from its surface form (unlike the non-inherent grammatical aspect). Lexical aspect divides verb into *states* and *events*, and usually further subdivides events into subclasses. These categorisations are "based on their behaviour in a variety of syntactic and semantic frames that focus on their features" (Dorr and Olsen 1997: 152), i.e. which lexical aspect a verb has is typically decided by watching how it can be used in certain contexts, depending on the feature set used. What these features are and what subcategories are appropriate descriptions is a contentious area in the field of theoretical linguistics, and also depends on which area of differences between verbs a researcher wants to highlight. Dorr and Olsen (1997) are of course interested in the elements of verbs that relate to temporality[4] and consequently

[4]Other areas of interest might be the complexity of the event expressed in a verb, or if a change of state is involved etc.

use the features dynamicity [$\pm d$], telicity [$\pm t$] durativity [$\pm r$] (called atomicity [$\pm a$] in (Dorr and Gaasterland 2007)), and the subcategories *activity, accomplishment* and *achievement*. The mapping of feature distribution to lexical aspect is pictured in Table 9.

Lexical Aspect	Telic	Dynamic	Durative	Examples
State			+	*know, have*
Activity		+	+	*march, paint*
Accomplishment	+	+	+	*destroy*
Achievement	+	+		*notice, win*

Table 9: Featural Identification of Lexical Aspect (Dorr and Olsen 1997: 153)

The basic distinction is between states and events, i.e. events have dynamicity whereas states do not. Durativity specifies whether events have a duration, and telicity applies if an event has an inherent goal, like *destroyed* in Example (6). *March*, on the other hand, is atelic.

(6) The soldiers destroyed the bridge in an hour.
* The soldiers marched in an hour.

But the lexical aspect of a verb is not only a function of the verb itself, it can change with the context in which it appears. Dorr and Olsen (1997) employ a so-called "privative" account of lexical aspect features: Unmarked features can become marked through additional elements in the sentence, but not the other way around: Marked features cannot become unmarked[5] For example, the atelic verb *march* can be coerced into a telic one by introducing a path to the sentence (Dorr and Olsen 1997: 152):

(7) The soldiers marched.
The soldiers marched to the bridge.
The soldiers marched to the bridge in an hour.

Consequently, to compute the lexical aspect, it is necessary to take into account the context in which the verb appears. For this purpose Dorr and Olsen (1997) compiled a dictionary of 10,000 English verbs that encodes their meaning and the sentences that they can engage in in a "lexical conceptual structure" (LCS). The LCS for the examples in (7), i.e. atelic and telic *march*, are given in Figure 2. The algorithm for extracting the lexical aspect from its LCS is given in Figure 3 and explained in Section 4.2.4.
Dorr and Gaasterland (2007) condense the temporal information conveyed by the lexical aspect class, using dynamicity [$\pm d$], telicity [$\pm t$] durativity [$\pm r$] (called atomicity [$\pm a$] in (Dorr and Gaasterland 2007)), in combination with the non-inherent grammatical aspect feature progressiveness [$\pm p$]. they determine if a verb describes an interval or a

[5] For a more detailed account of this particular theory, see (Olsen 1994). For a different account of how coercion works and which types of coercions are permissive and which are not, see Moens and Steedman (1988).

```
(act loc (* thing 2) (at loc (thing 2) (thing 11))
       (march+ingly 26))

(go loc (* thing 2)
          ((* from 3) loc (thing 2) ([at] loc (thing 2) (thing 4)))
          ((* to 5) loc (thing 2) ([at] loc (thing 2) (thing 6)))
```

Figure 2: Lexical Conceptual Structure for atelic and (coerced) telic *march*

point event: They assume an Event E to have a starting time E_s and a finishing time E_f:

(8) a. He winked.
b. He laughed.
c. He knew Spanish.

In a point event, start and finishing time are the same as in (8a). A closed interval has an endpoint as in (8b). An open interval has no endpoint as in (8c). Note that all examples in (8) are in the past simple tense.
The inclusion of the progressiveness feature is necesseary, because point events can be coerced into interval events by the progressive tense, as evidenced by the examples in (9) (cf. Gisborne 2010: 17):

(9) a. I dropped a pencil.
b. I was dropping a pencil.

The event described in (9a) is a point event that takes place once. The progressive tense in (9b) coerces the event into an interval event, by making the event into a repeated action. Or, in the terms of Dorr and Olsen (1997), the use of the progressive changes the unmarked durativity feature to marked.
So in this account of lexical aspect, we can see how lexical and grammatical aspect are related: Progressive aspect marks a non-durative verb as durative, and perfect aspect marks an atelic word as telic. Consequently, the incorporation of the lexical aspect in this form provides us with an additional feature that is closely related to that of tense and grammatical aspect and is capable of filling the void of unmarked (i.e. simple) within-tense combinations.
They made a thorough analysis of all possible Allen relations that can be expressed by a temporal connector depending on the tense and aspect of the matrix and adjunct verb through analytical – not empirical – analysis, and sorted their findings into tables according to permissible tense-combinations, sorting them within the tables according to the interval of the matrix and adjunct verb. The interval expresses if the event expressed by the verb has duration based on the lexical aspect of the verb. The mapping of lexical aspect to intervals is pictured in Table 10.

Interval	Lexical Aspect			
open	[-d]	[±t]	[±a]	[±p]
	[+d]	[-t]	[±a]	[+p]
point	[+d]	[±t]	[+a]	[-p]
	[+d]	[+t]	[+a]	[-p]
closed	[+d]	[+t]	[-a]	[±p]
	[+d]	[-t]	[±a]	[-p]

Table 10: Intervals corresponding to aspectual features

Mat/Adj	AFTER	WHEN	WHILE
C/C	>	oi s si d f >	= o oi s d f
C/O	oi mi f	= oi s si d mi f	= o oi s d f
C/P	>	oi s d f >	= o oi s d f
O/C	>	= o oi s si d di m mi f fi >	= o oi s d f
O/O	oi mi f	= oi s si d di mi f	= o oi s d f
O/P	>	oi s d f >	= o oi s d f
P/C	>	oi s si d	= s d f
P/O	oi m f >	oi s si d >	= s d f
P/P	>	= oi s d >	= s d f fi

Table 11: Possible Allen Relations for past/past matrix/adjunct verbs depending on their interval according to (Dorr and Gaasterland 2007: 1694)

4 Implementation

4.1 Extraction of Rules

In order to discover disambiguating rules, the Wall Street Journal section of the Penn Treebank corpus was analysed for tense, grammatical and lexical aspect. These features were selected for several reasons: The work in Dorr (1992a), Dorr and Olsen (1997), Dorr and Gaasterland (2002) and Dorr and Gaasterland (2007) gives plenty of reason to assume that tense, grammatical and lexical aspect can serve as distinguishing features in this particular disambiguation task, while also laying an excellent foundation for the exploitation of said features in this task. Tense and grammatical aspect (this was called Temporal Signature in Lapata and Lascarides (2006: 94-95), see also Section 3.2; the combination of tense, grammatical and lexical aspect will be called Temporal Signature from here on out.) wasn't a very useful feature in the original system, but in their classifier it was supposed to be a predictor of temporal connectors, and they can occur with every possible Temporal Signature combination. Within those temporal connectors, on the other hand, it is reasonable to believe that the temporal signature is a distinctive feature for temporal relations in conjunction with temporal connectors. One need only look at the examples presented in Dorr and Gaasterland (2002: 39-41), which feature the same verbs and connectorswith varying are tenses and aspects, to see that these are distinguishing features in this task. And lastly, the decision to avoid

features that are used in the classifier is a concious one to avoid bias towards a feature that was used for disambiguation in the classifier.
The decision to base the ruleset on a corpus analysis rather than theoretical linguistical assumptions is the same as the one behind corpus linguistics in general: theoretical assumptions that are based on anecdotal and simplified data can never represent the full picture of language use, and considering only toy examples likely leads one to overlook phenomena that do not immediately spring to mind. In the other extreme, it can lead one too put to much significance on occurrences that are rare.
A drawback of the empirical approach is that it might be too narrow, i.e. only applicable to a certain domain, because every text type has its own set of idiosyncrasies. In this case, the domain might be as small as the Wall Street Journal, although, more likely, the domain in this case is "news articles." To find out, the disambiguator would have to evaluated on a different domain, which is currently not possible. On the other hand, if one operates within-domain, this can be an advantage, because the rules extracted are hand-tailored for this domain, and also because it eliminates the need to take all possibilities of language production. The more codified the domain, the easier it becomes to describe.
Two starting assumptions were made: Hypothetical events, usually marked with a modal verb, were not removed, even though the events they express are not actual events, but are situated in another possible world – for which the SLink provides an annotation tag – with the reasoning that hypothetical events, at least with regard to the feature set used in the disambiguator as well as the classifier, do not behave differently from actual events, and can consequently be used for training.
The second one concerns the treatment of verbs that fall under the ALink, i.e. words that mark a beginning or ending of another event (its argument) were treated as automatically classifying their argument event as either BEGINS or ENDS respectively.

4.1.1 Rules based on temporal signature

After, before, until: These connectors are not disambiguated: The analysis of Dorr and Gaasterland (2007) described in Section 3.4 suggests that the relationship expressed by these connectors is relatively consistent: Accordingly, *after* and *before* were mapped to the BEFORE relation, and *until* was mapped to the ENDS relation. Additional Allen relations for *before* according to Dorr and Gaasterland (2007: 1694) are 'overlaps' and 'finishes inverse' (apart from 'meets' which is folded into the BEFORE relation anyway), 'overlaps inverse' and 'finishes' for *after*. The possible relations given for *until* are 'meets' and 'starts', and for *since*, the possible relations given are 'meets', 'after', and 'finishes', which is odd, because this is the inverse mapping to Lapata and Lascarides (2006: 108). There seems to be a different understanding of what these relations mean between the two papers. Table 3 pictures the mappings as presented in the respective papers.

Once: Several relations can be encoded using *once*. The examples in (10) show different temporal signature combinations that are significant:

(10) a. Although the network isn't connected to the computer systems that operate either Galileo or the shuttle, part of the network *will carry analyses* of Galileo data **once** the craft *gets spaceborn*.

b. **Once** this plan *is approved*, Tan Sri Basir said, most of Bank Bumiputra's nonperforming loans *will have been fully provided for*.

c. The Oakville plant *could face losses*, **once** the additional car-making capacity across North America *is operating*.

d. Typically, developers option property and **once** they *get* the administrative approvals, they *buy* it.

The combination Future Simple + Present in (10a) marks a BEGINS relation: The main clause event will begin when the subordinate clause event starts. Progressives also mark a BEGIN relation, as in (10c): An ongoing event starts another event. This is also true for progressives in the main clause, in that case the subordinate event sets off the main clause event. The combination Future Perfect + Present in (10b) marks a ENDS relation: The main clause event will be achieved when the subordinate event ends. The default for *once* is the BEFORE relation as in (10d): the subordinate event is a precondition to the main event.

Since: This connector is difficult, because it can be used as a purely causal connector. Dorr and Gaasterland (2007: 1694) allege, that *since* does only convey a temporal meaning in certain interval combinations, essentially all combinations that involve an open interval, but examination of the corpus data did not confirm this. All of the instances in the development set with open intervals for *since* were stand-alone "be" as in Examples (11a,b,d,e), only one of which does not denote a temporal relation, namely (11a).

(11) a. However, a Canadian Embassy official in Tel Aviv said that Canada *was unlikely to sell* the Candu heavy-water reactor to Israel **since** Israel *hasn't signed* the Nuclear Non-Proliferation Treaty.

b. **Since** NBC's interest in the Qintex bid for MGM/UA *was disclosed*, Mr. Wright *hasn't been available* for comment.

c. The 486 is the descendant of a long series of Intel chips that began dominating the market ever since IBM picked the 16-bit 8088 chip for its first personal computer.

d. The bond issue *is* TVA's first public offering **since** the Financing Bank *was created* in 1974, primarily to finance the TVA.

e. In Brussels, it *was* the first trading day for most major shares **since** stocks **tumbled** on Wall Street Friday .

If it appears with a perfect tense in the subordinate clause (Example (11a)), it is in most cases not used in a temporal sense. A perfect tense in the subordinate close on the other hand marks a BEGINS relation (Example (11b)). So does a point event in the subordinate or main clause as in Example (11c). Example (11d) shows a disallowed tense combination (present-past) that is not uncommon with the *since* connector: It orders an event into an interval that has an event as its starting point. The event that is the starting point of the interval specified by it, however, is completed before the event in the main clause takes place – it only serves as a reference point in the past. The relation that applies here is *before*. Dorr and Gaasterland (2007: 1698) term this behaviour *Extended past up to present*, it "enables coherent rendering of an enduring state that leads up to the starting point of a current event." which is particularly often the case in connection with *since* and consequently, CDTS violations in conjunction with *since* are not reanalysed in the disambiguator. Essentially the same use of the connector *since* is pictured in Example (11e): a simple state tense (auxiliary + infinitive) in the main clause indicates use of *since*: anchoring an event on a timeline by referencing a past event. Senteces like these were consequently marked as BEFORE. *Since* also rarely appears with progressives, at least there were no instances in the development corpus. Because of this, all instances of *since* with progressive tense were dropped.

As, while, when: These connectors have some rules in common, likely because they primarily engage in overlapping relations. Progressives combined with other tenses/aspects designate an event that includes the other event if the other event is a point event, or are simultaneous if the other event is an interval event. In these instances, the subordinate event serves as a reference frame for the main clause event.

(12) The proposed merger *comes* **as** K-mart's profit *is declining* and sales at its core discount stores *are rising* more slowly than at such competitors as Wal-Mart Stores Inc.

(13) Mr. Dinkins did fail to file his income taxes for four years, but he insists he voluntarily *admitted* the "oversight" **when** he *was being considered* for a city job.

(14) **While** she *is wondering* whether to live it up, and do something even more dramatic, say get married, her life *is further complicated* by the reappearance of an old flame, David, a film critic and actor who always seems to be just on the brink of stardom.

As: A perfect tense in the main clause describes an event that was set off by the event in the subordinate clause. The subordinate event might still be ongoing.

(15) a. A number of those polled predict the dollar *will slip* **as** the Federal Reserve *eases* interest rates.

b. And he noted that even if there aren't any reductions in the current C-5B contract payments, Lockheed *will lose* that business in the next few years **as** the last of the planes *are delivered.*

c. A widening of the deficit, if it were combined with a stubbornly strong dollar, would exacerbate trade problems – but the dollar *weakened* Friday **as** stocks *plummeted.*

d. Prices of Treasury bonds *surged* in the biggest rally of the year **as** investors *fled* a plummeting stock market.

e. Recently, a contractor *saved* her from falling three stories **as** she *investigated* what remained of an old Victorian house torched by an arsonist.

The combination Future-Present in Example (15a) marks the event in the subordinate clause as the trigger for the main clause event: The temporal relation here is BEGINS. In Example (15b), the future tense is paired with a stative event. In these cases, the subordinate event marks the ending condition of the main clause event.
Simple tenses are sorted depending on their interval pairing: if both intervals are the same (Example (15c)), the relation is SIMULTANEOUS, if a point event is involved in the main or subordinate clause (Example (15d)), the point event often serves as a trigger for the other event, so sentences involving point events and interval events belong to the BEGINS relation. Sentences involving mixed intervals are belong to the INCLUDES relation (Example (15e)).

When:

(16) a. If it finds one and gets into the system, it *will display* a screen **when** a user *logs on* that says, " Worms Against Nuclear Killers ...".

b. Atsushi Muramatsu, Nissan's executive vice president for finance, *helped* set the tone in December 1986, **when** the company *was heading* toward the first operating loss by a Japanese auto maker since the nation's postwar recovery.

c. Periods before the advent of futures or program trading *were often more volatile*, usually **when** fundamental market conditions *were undergoing* change.

Mixed tense (future - present) signals a BEFORE relation, as in Example (16a) Mixed grammatical aspect (progressive - simple, simple - progressive) denotes an INCLUDES relation (Example (16b)), or an SIMULTANEOUS, if the adjunct event is an interval (Example (16c)), but BEFORE relations are also possible in that combination. It might be an worth excluding this combination, if the results become too noisy. Perfect aspect is not a very reliable predictor of the temporal relation

(17) a. Technically, Mr. Kaiser noted that a lot of traders **had bought** into the market **when** the price *was* in the $1.24 to $1.26 range, thinking there was support at the $1.20 level.

b. The young John Gutfreund *had been discovered* by Billy Salomon of Salomon Bros. **when** he *was* still a bearded liberal, and put to work as a trader, and then as a rough-and-tumble syndicator.

c. Essentially, Mr. Freeman *had invested* heavily in the Beatrice leveraged buy-out, **when** he *was* told by another prominent trader, Bernard "Bunny" Lasker, that the deal was in trouble.

d. **When** they *return* to their desks at 1 p.m., they *have pedaled* 20 miles.

Examples (17a-b) show an INCLUDES relation, whereas (17c-d) show an BEFORE relation. The additional rule described in the next section doesn't apply here, for Examples (17a,d) have different subjects, and in Examples (17b,d), the subjects in main and subordinate clause are identical. See Section 4.1.2 for an explanation. Since this distribution is roughly equal, this tense combination was deemed unpredictable and left out.

For simple - simple pairings, intervals are not sufficient for predicting temporal relations, additonal disambiguation features needed to be found. These are described in the next Section.

4.1.2 Additional Rules

An effort was made to discover rules that are more robust for disambiguation than temporal signature, for the temporal connectors *when* and *while*, this effort was successful: the discovered rules both belong to the realm of discourse relations.

(18) a. **When** *she* met the local press for the first time on Friday, *Mrs. Hills* firmly reiterated the need for progress in removing barriers to trade in forest products, satellites and supercomputers, three areas targeted under the Super 301 provision of the 1988 trade bill.

b. But **when** *the company* revealed Lisa's poor sales late in 1983, *the stock* plummeted to a low of $ 17.37 a share, according to the suit.

When is by far the most used temporal connector, in all corpora. It makes up for almost 50% of all sentences containing temporal connectors. It also the most diverse of all the temporal connectors: Potentially, it can express any temporal relation (cf. also Table 11). Consider the following examples from Moens and Steedman (1988: 15):

(19) When they built the 39th Street bridge,

a. a local architect drew up the plans.

b. they used the best materials.

c. they solved most of their traffic problems.

In this example, *when* serves to express a BEFORE relation in (19a), an INCLUDES relation in (19b) and an AFTER relation in (19c), and all examples are in the past simple tense, so temporal signature is not enough for disambiguation.
In attempt to further disambiguate, it is worth looking at the context in which the connector appears: Looking at the corpus, it seems like identity of subject plays a role: If the subjects of main and subordinate clause are identical, as in Example (18a), the subordinate clause usually sets a reference time to when the event in the main clause happened, i.e. the main event is included in the subordinate event. In opposition, if the subjects differ, as in Example (18b) usually one event brings about the other, i.e. the main event happens after the subordinate event.
This is not a perfect predictor, Example (19c) offers a counter-example, but in most of the cases, this was a good predictor.
Additionally, for same-subject sentences, if the events have matching intervals, their relation is SIMULTANEOUS, if not, their relation is INCLUDES.

(20) a. Domestic items *fell* 29% , **while** re-exports *rose* 56%.
b. *One* stuck to old-line business tradition, **while** *the other* embraced the change.
c. Euro Disneyland shares *made a debut like Snow White* yesterday **while** most of the London stock market *looked like it had eaten the Evil Queen's poisoned apple.*
d. **While** coal is abundant and cheap, it is also polluting.

While is often used in contrasting contexts, in fact, it can be used in a purely contrasting way as in (20d). Lapata and Lascarides (2006: 94) state that the percentage of non-temporal use of *while* in the BLLIP corpus was 13.3%. They estimated that number by randomly selecting 30 examples from the corpus. It can be argued that even in these contexts, *while* retains a temporal connotation, if ever so slightly. In any case, *while* can be used to contrast two events in a temporal way, i.e. two events are described to take place simultaneously. To test for a contrasting context, we can test for synonymity/antonymity of verbs. Examples 20 a. - b. illustrate a variety of contrasting contexts that express simultaneity, with a. showing a contrast expressed in the antonymity relation between the verbs, b. expresses the contrast in subjects and c. expresses a contrast that can probably not be recognised without extensive world knowledge.
Testing for contrasting subjects is not an easy task, for contrasting subjects rarely are in an antonymy relationship – the contrast much rather springs from the relationship they have with each other, which is usually only available through world knowledge, e.g. the opposition of "Euro Disneyland shares" and "most of the London stock market" is not easily derivable. Only if contrasting pronouns are used as in (20b) is the opposition easily derivable.

4.1.3 The Ruleset

An overview of all the rules that are applied for disambiguation and the lisp-code of the function implementing the rules are given in the appendix. Note that in the code, *while* and *when* have already been disambiguated, the results are read in at the start of the function. Also, for *as* and *when*, a function checks beforehand if the matrix verb (the verb of the main clause) or the adjunct verb (the verb of the subordinate clause) belong to a predefined set of words signalling 'start' or 'end'. If they do, they are automatically sorted into the respective relations before the function is called.

4.2 The Algorithm

In this section, I will first outline the algorithm of the disambiguator and then discuss every step in a dedicated subsection.

1. **Prepare the data:**
   Extract SBAR-TMP nodes and make the data lisp-readable.
2. **Extract matrix and adjunct verbs:**
   Find the VPs that correspond to the main verbs of main and subordinate clause.
3. **Infer tense and grammatical aspect:**
   Using a pattern matcher, infer the tense and grammatical aspect of the extracted VPs, reanalyse if a clash is found.
4. **Infer interval:**
   Infer the corresponding intervals for matrix and adjunct verb.
5. **For temporal connector *when*:**
   Find subject nodes, run coreferencer on sentences.
6. **For temporal connector *while*:**
   Determine if verbs are in a synonymity/antonymity relationship.
7. **Apply rules:**
   Sort sentences into TimeML relations according to set of rules.
8. **Train the classifier:**
   Use the preprocessed training data to train the classifier.

4.2.1 Preparing the data

To avoid analysing sentences whose matrix/adjunct verbs do not have a temporal relation, the first step is to extract only sentences that contain a SBAR-TMP node, signalling a temporal relation. The fact that the Bllip corpus is automatically parsed introdues some noise into the data. Also, certain characters have to be escaped to make the sentences lisp-readable.

4.2.2 Extracting main and subordinate verb

The pattern-matcher was taken from Dorr and Gaasterland (2007)[6] and adjusted for use with the Bllip Corpus. Furthermore, there was a mistake in the pattern-matcher that prevented composite tenses containing an infinitive such as "will be" and "did <infinitive>", because it excluded VB (= infinitive nodes). This was because the pattern-matcher prefers the VP closest to the temporal connector, and if the pattern-matcher allows VB nodes, this may lead to the algorithm collecting meaningless VP nodes. The exclusion of VB nodes for the Bllip is valid, since it uses AUX for auxiliaries and not VB like Penn Treebank.

4.2.3 Inferring tense and grammatical aspect

The extracted VP nodes are then matched against patterns these had to be adjusted to work with the Bllip corpus, since it uses an AUX tag for auxiliary verbs, which Penn Treebank doesn't. If there is a clash found, (i.e. a past-present combination), tenses are reanalysed in the following ways: For *since*, clashes are allowed (see Section 4.1). If the present tense verb is a modal, mark it as past simple. If none of these apply, try to find a VP node above the one currently used. If reattachment doesn't succeed, the sentence is dropped. Clashes involving perfect aspect of either tense are not reanalysed, since (Dorr and Gaasterland 2007: 1698) identify these as *one-way causal relationships*, in which the subordinate event causes the event in the main clause. These sentences are automatically dropped.

4.2.4 Inferring interval

The pattern matcher was taken as the basis. It was enhanced with an algorithm to deduce the non-inherent lexical aspect, or rather to deduce intervals based on lexical aspect, combining the approaches described in Dorr and Gaasterland (2007) and Dorr and Olsen (1997). To deduce an interval (see Section 3.4), several steps are taken: First, the lemmatised verb form is looked up in an lexical conceptual structure (LCS) dictionary[7]. If there are several options, the algorithm sees if they all produce the same interval. If this is not the case, the argument structure as given by the file is matched against the full VP of the verb in question: The first argument is filled by the subject. The full VP is then searched for NP nodes (direct objects) and PP and TO nodes (indirect objects). All available prepostions within the VP node are extracted and saved as the argument structure of the verb in question. This structure is then matched with the argument structure and the longest match is then returned, allowing for partial matches where not all positions are filled.
From the returned LCS structure, the lexical aspect is then derived using the algorithm from (Dorr and Olsen 1997: 156) pictured in Figure 3: The top node is examined for one of the telicity indicators CAUSE, LET, GO. If one is found, the verb is marked

[6] Available at `ftp://ftp.umiacs.umd.edu/pub/bonnie/CDTS-Solution-2006.lsp`
[7] Available at `http://clipdemos.umiacs.umd.edu/englcslex/download.html`

as telic. CAUSE and LET also signal dynamicity and durativity, GO only dynamicity. If none is found, the top node is checked for atelicity indicators ACT, BE, STAY. If one is found, it is checked whether on of the inside nodes contains one of the coercion markers TO, TOWARD or FOR_{TEMP}, if one is found, telicity is marked. BE, STAY indicate durativity, ACT indicates dynamicity and durativity. The set of features is then returned and the corresponding interval is then looked up in a table like Table 10.

initialize T(L):= [∅T], D(L):=[∅D], R(L):=[∅R]
if Top Node L ∈ {CAUSE, LET, GO} **then**
  T(L):=[+T]
  if Top Node of L ∈ {CAUSE, LET} **then**
    D(L):=[+D], R(L):=[+R]
  else if Top node of L ∈ {GO} **then**
    D(L):=[+D]
  end if
end if
if Top Node of L ∈ {ACT, BE, STAY} **then**
  if Internal Node of L ∈ {TO, TOWARD, FOR_{Temp}} **then**
    T(L):=[+T]
  end if
  if Top Node of L ∈ {BE, STAY} **then**
    R(L):=[+R]
  else if Top node of L ∈ {ACT} **then**
    D(L):=[+D], R(L):=[+R]
  end if
end if
return T(L), D(L), R(L)

Figure 3: Algorithm for Lexical Aspect Determination

4.2.5 *When* connectors

After the temporal structure has been determined, the sentences are sorted by their temporal connectors into two tables, one corresponding to time intervals and one to tense and grammatical aspect, i.e. every sentence is sorted into two disjunct matrices, one representing its tense combination and the other its interval combination. From there, the rules are applied to sort them into TimeML relations.
For the temporal connectors *when* and *while*, additional processing has to be done. To sort the *when*-sentences, a coreference resolver is applied to determine if the subjects of matrix and adjunct verbs co-refer. The coreferencer used for this is part of the Stanford CoreNLP package[8]. Pretrained coreferencers usually work on raw text, which means that they re-parse the already parsed input. This is unforunate and should be avoided by implementing a dedicated coreferencer into the system, but this was not within the scope of this work. Also, a coreferencer is necessarily more accurate when it is run on a whole text instead of just a single sentence, but in view of the available time,

[8] `http://nlp.stanford.edu/software/corenlp.shtml`

the choice was made to run it only on isolated sentences. The coreferencer outputs an XML file which was parsed using python and all sequences tagged as co-referring were compared to the NP-SBJ nodes of the matrix and adjunct sentence, which relies on the parser to identify the same sequences as subjects, another reason to prefer a dedicated coreferencer that doesn't re-parse.

4.2.6 *While* connectors

To determine the synonymity or antonymity of the *while*-sentences, the WordNet[9] API from the Natural Language Toolkit[10] is used. To find out which sense of a lemma we need to compare, the same argument matcher is used that was used to find the corresponding LCS, only this time it returns one or several WordNet sense keys. Since these keys are different in each version of WordNet, WordNet version 1.6 was used, because the LCS lexicon used contained only references to WordNet 1.5 and 1.6. The results are fed into WordNet, which looks up the ANTONYMS() and SIMILAR_TOS() for each sense key and sees if there is a non-empty intersection between the returned antonyms/synonyms of the matrix verb and the sense key of the adjunct verb. The algorithm acts greedy, in that it compares all lemmas in the synsets, since the antonyms and synonyms in WordNet do not include "weak" antonyms like *rise* and *fall*, but only "strong" antonyms like *increase* and *decrease*. It might have been better to use a similarity function as provided by the WordNet API, since antonyms should still have a higher similarity rating than completely unrelated words, because they still belong to the same domain. This could be used to make a reverse search, i.e. excluding words that are below a certain threshold, which would have to be determined in some way. However, in this approach, the similarity measure wasn't used.

4.2.7 Applying the rules

Once all the extra rules are applied, the sentences are sorted into their corresponding TIMEML relation: The algorithm iterates through the tensetable and the rules specified in Section 4.1 are applied. If interval is part of the rule, intersection or disjunctions with the interval-table are generated. The full algorithm is attached in the appendix, together with the corresponding lisp code.
After everything is sorted, the last step is to retrieve the fully parsed sentences, for we have only been dealing with sentence numbers so far, and produce files with the preprocessed training data.

[9] `http://wordnet.princeton.edu/`
[10] `http://www.nltk.org/`

5 Evaluation

5.1 Disambiguator only

In the first evaluation, the diambiguator was tested on its own to verify if the right assumptions have been made regarding the set of rules that was used.
The disambiguator was tested on TimeBank by taking the Wall Street Journal articles from Penn Treebank, and acquiring raw text files of the other data in the corpus. These were parsed using the Charniak Parser[11]. In the case of Penn Treebank, only SBAR-TMP nodes were considered, in the other case all parses were fed into the disambiguator, which probably led to some sentences being overlooked by the disambiguator due to false parses. Because event annotation wasn't part of this project, the output was verfied using the `signalID` of the connector in the corresponding TLinks. If the connector had no `signalID`, the temporal relation of the nearest events to the left and right of the connector was chosen. If there weren't any, the relation couldn't be verified ('no relation' in Tables 12 and 13.)

	correct	incorrect	no relation	Total
before	11	0	0	11
after	28	1	0	29
until	9	6	1	16
when	13	7	4	24
while	1	0	2	3
as	6	2	0	8
since	4	2	0	6
once	3	0	0	3
Total	76	18	7	100

Table 12: Disambiguation results by connector

	correct	incorrect	no relation	Total
before	53	4	0	61
includes	5	5	6	12
simultaneous	7	1	0	8
begins	1	2	0	3
ends	9	6	1	16
Total	75	18	7	100

Table 13: Disambiguation results by relation

The total number of connectors tested doesn't match up with the numbers given in Table 5, not only because some parses were not valid, but also because not all temporal connectors in TimeBank have a SIGNAL tag, and secondly because not all temporal connectors with a SIGNAL tag in TimeBank do actually connect two sentences, for instance, *since* and *after* are often used as prepositions.

[11] `http://www.cs.brown.edu/~ec/\#software`

The results are shown in Tables 12 and 13: Table 12 shows the disambiguated connectors and whether they were disambiguated correctly, incorrectly or if no TLink was available. Table 13 shows the temporal relations and whether they were correctly identified, whether they were incorrectly applied, and what label was applied to non-existant relations.

The test data isn't very meaningful, because the numbers are so small, and because 56% of connectors were not disambiguated (*after*, *before* and *until*). Still, we can glean from the results that in the case of *after* and *before*, it was the right decision not to disambiguate. In the case of *until*, it might be worth considering to disambiguate this connector as well, but it is almost the only source of training data for the ENDS relation. On the other hand, higher quality training data might be preferrable to more training data.

The numbers for *when* are promising: a majority of the disambiguated cases agreed with the data, even though the coreferencer prefers the BEFORE relation, which is unfortunate and should be remedied with a dedicated coreferencer that doesn't re-parse. It is worth considering additional rules that might help disambiguate *when* further after coreference resolvement.

Regarding *while*, there isn't much to conclude from the data, and the one correctly identified *while*-sentence contained a perfect synonym, i.e. the same word in both main and subordinate clause and was correctly identified as SIMULTANEOUS.

The BEGINS relation is difficult, it is a rare relation in TIMEBANK in general, and there is not much in terms of tense and lexical aspect to discern a BEGINS relation from an INCLUDES relation. Including trigger words, i.e. words that signal a beginning is really the only clue. The same can be said about ENDS.

Once disambiguation seems to work perfectly, although none of the cases were cases where any of the rules applied that deviate *once* from the default interpretation, but on the other hand, there weren't any wrongly disambiguated instances either, so we can say that the rules for *once* are working as expected.

As and *since* are mostly correctly disambiguated, *since* was labelled as INCLUDES, when it should have been SIMULTANEOUS, and the same is true for *as*. This has probably to do with the interval feature, either verbs are given the incorrect interval by the disambiguator, or the annotators might consistently prefer SIMULTANEOUS over INCLUDES, when the timelines are approximately the same, regardless of the actual time intervals involved, i.e the disambiguator might be overly specific.

In conclusion, these results are rather good, considering that some of the rules apply only to the majority of cases, so there is some noise to be expected. Additionaly, the events that the `signalID` refers to are not necessarily the verbs on which the disambiguation was based, so a few disagreements are to be expected because of this as well.

5.2 Disambiguated training data

The disambiguator was then used to preprocess the training data for the Lapata and Lascarides (2006) classifier. The results are shown in Table 14. Table 15 shows the number of training instances.

	not preprocessed		preprocessed	
TLink	Accuracy	F-score	Accuracy	F-score
BEFORE	46.4	47.6	47.3	48.5
BEGINS	10.5	7.8	11.2	10.3
ENDS	14.1	3.7	14.1	5.6
INCLUDES	50.0	51.5	52.6	53.4
SIMULTANEOUS	46.7	47.8	48.9	51.2

Table 14: Evaluation Results

TLink	not preprocessed	preprocessed
BEFORE	31,643	20,270
BEGINS	2,810	3,447
ENDS	5,333	5,344
INCLUDES	21,859	23,441
SIMULTANEOUS	22,165	6,583
Total	83,810	59,085

Table 15: Number of training instances

The raw data wasn't available, and so no significance test could be performed. The results for the preprocessed training set are slightly better, even though the new training set was only 70% the size of the original training set. In this regard, the improved result for the SIMULTANEOUS relation is especially remarkable, since the number of training instances for this relation was only 30% the size of the original number. But overall, the results are rather disappointing.
There are several possible explanations for this:

1. The disambiguation was not successful
2. The feature set from the original system is not the best for this task
3. The data is too diverse after disambiguation
4. A combination of the above

Ad 1) Possibly the disambiguator didn't do a very good job. As can be seen from Table 15, a lot of sentences (30%) were dropped by the disambiguator, either because it couldn't infer a tense or lexical aspect, because none of the rules applied, or because a CDTS conflict couldn't be resolved. Maybe the data produced was still too noisy to produce better results: This is a strong possibility, since many of the rules were not strictly dismabiguating.

On the other hand, the evaluation of the disambiguator on its own was not too bad. If it indeed did a bad job, the results should have been worse due to lesser training data. In addition, the results are very similar to the original results, and even though the data was from the same corpus, it was split among the relations in a different way, which should have lead to more differing results, especially in the case of BEGINS, because the training data used for this relation was entirely different, (i.e. no exclusive mapping of *since* to this relation.) Admittedly, the results for BEFORE and ENDS especially were to be expected, ENDS was trained on essentially the same set. BEFORE was somewhat more diverse, but the majority of this training set is probably comprised of *before* and *after* sentences, as it was in the original system. This suggests that the feature set used in the classifier is somehow biased. The similarity of the results is more difficult to explain than simply better, or worse for that matter, results would be.

Ad 2) It stands to reason that the most prominent feature in the ensemble model, syntactic signature, is not a very useful, or maybe somehow biased feature when it comes to annotating pure temporal relation. Lapata and Lascarides (2006: 104) state:

> Our results so far [...] indicate that the syntactic complexity of the two clauses is another key predictor. [...] Soricut and Marcu (2003) find that syntax trees are useful for inferring discourse relations, some of which have temporal consequences.

It seems that syntax trees are less useful in inferring temporal relations than they are in inferring discourse relations – which makes a lot of sense, after all, when inserting discourse markers into text, the syntactic structure is pre-formed to house a specific discourse marker (especially in a cloze task), whereas several different temporal discourse markers can express the same temporal relation and be syntactically diverse.
The performance of both versions of the annotator regarding the BEGINS and ENDS relation was explained with the comparably small number of training instance in Lapata and Lascarides (2006: 109), but the equally small set of instances for the SIMULTANEOUS relation in the preprocessed version casts doubt on this assumption.
Most likely, these two relations are not sufficiently captured through the features used: As has already been pointed out in the previous section, these relations are also difficult to capture with the features that were used in the disambiguator.
The training set for ENDS was essentially the same as the one in the original system and consequently scored the same. As could be seen from the last Section, *until* on its own is not a good indicator for this relation.

Ad 3) This is related to the last point, i.e. that the feature set is not capable of capturing the idiosyncrasies of the data set now that the variance in the data is bigger, because of the presence of a variety of temporal connectors with their specific syntactical pecularities.

Ad 4) Lastly, it is of course entirely possible, that both systems were not up to the task, although, in that case, we would have expected worse results.

6 Conclusion

6.1 Discussion

The project was about trying to disambiguate temporal connectors to use them as indirect markers of temporal relations in text, to propel the use of unsupervised learning techniques in the field of temporal semantics.
Disappointingly, the overall outcome of this project is rather inconclusive: The first evaluation isn't particularly meaningful with such a small test set, and the outcome of the second evaluation is surprising at best. It can't really be decided if the disambiguator is working as expected, or if it could contribute to unsupervised learning of temporal relations. A first step to test this would be to test different feature sets in the classifier and see if they produce significantly different results for both the preprocessed and the unprocessed data set, i.e. essentially readjusting the features to the new task. After all, the best feature sets were determined not for the temporal relation annotation task, but for the insertion of temporal connectors task, and only then applied. It is reasonable to think that these two task have different requirements regarding the features that are useful in discerning them, especially for BEGINS and ENDS, which show the most room for improvement.
Mani et al. (2006) report an F-Score of 45.16 for BEGINS without transitive closure, and 83.87 with transitive closure, using only the features available in the TIMEML EVENT tag. The question is, which of these features are important to the relation and how to integrate them into either the classifier or the disambiguator.
Chambers et al. (2007) don't report their results splitted by temporal relation, but it would be interesting to know if any of the additional features, which were mainly taken from Lapata and Lascarides (2006), were useful in the classification of the BEGINS relation.
In an application, the question is how useful those relations really are. Allen states that he included further subdivisions to the *during* relation, because "it provides a better computational model." (Allen 1983: 834)
From an NLP point of view, it might be advantageous to only include the subdivions of *during* when they are necessary, e.g. in a recognising textual entailment task, where it is necessary to reason over events and their relation to each other, the inclusion of BEGINS and ENDS might make it easier to entail non-temporal relationships between events. In an ordering task,the exact nature of an inclusion relation is irrelevant.
It would be desirable to have a set of relations that can be agreed upon by the community, but this isn't very likely, on the on hand because of different requirements regarding what kind of relations are relevant to a given task, and on the other hand regarding the granularity of our perception of time and the implications that come in a bundle with a temporal expression.

6.2 Improving the Disambiguator

The inclusion of the lexical aspect as an interval, at least in theory, is useful for circumventing the absence of an aspectual marker in the English simple tenses. The usefulness of the distinction between closed and open intervals, however, is not clear. It was also difficult to extract rules regarding intervals for connectors that were rare in the development set, because one example doesn't make a rule. Reviewing the data extracted from the BLLIP corpus would help extracting more rules for rare connectors.
One big problem with the disambiguator was that it relies on so many outside factors that can significantly harm performance, i.e. quality of parses, accuracy of the interval feature, accuracy of the coreferencer, the verb has to be in the LCS lexicon and in WordNet.
Something that could be done to improve disambiguation is finding additional features for the disambiguation of *when*: Many of the rules used for the disambiguation of *when* were not good predictors, but merely reduced the ambiguity in a set of *when* sentences. Since this is the most used and most ambiguous temporal connector, it is to be expected that more features would lead to improved results, in both the disambiguation task and the temporal relation annotation task, since most of the training data consists of *when*-sentences. Inclusion of some kind of world knowledge like EventNet might be useful, looking at the way events usually relate to each other.
While disambiguation did not work very well, using the functions provided by NLTK for the Synset class. Most of the words were not correctly identified as synonyms or antonyms. A more lenient measure would be more useful. An idea might be to use a similarity measure, which could also be applied to antonyms, since they usually belong to the same domain and are usually more related than unrelated words. A problem might be that words that appear together in a context probably are from a close domain, so setting a threshold for this measure might be difficult.

6.3 Evaluating the Disambiguator

The disambiguator itself was primarily designed to produce input for another system, so evaluation was primarily built around this. Testing on TIMEBANK was difficult, due to the small size of TIMEBANK and the even smaller number of sentences containing temporal connectors, not all of which express any temporal relation, or at least do not contain events annotated with TLINKS.
One evaluation method might be to present the disambiguated sentences and let them decide if the temporal relation is correct. But human annotators are expensive, and also not very reliable, although it might be easier to decide if a given relation is acceptable than to assign a relation. Another difficulty here is that no events were annotated, so the judgements might differ with regard to what constitutes the main event expressed in main and subordinate clause. There also isn't a baseline for this task. An easier method could be to modify the disambiguator and disambiguate for the relations in the TempEval challenge and test on their data, given that it contains sentences containing

temporal connectors. In any case, the biggest problem in evaluating the disambiguator is to find annotated data. While finding the rules, it became also apparent that it should be possible to disambiguate the unfolded relations, i.e. INCLUDES, IS_INCLUDED, BEFORE and AFTER, (the others are problematic, see above).
A thusly modified disambiguator might then also be useful in a sentence ordering task, although it has the disadvantage that it only works on temporal connectors.

6.4 Summary

The project described in this dissertation did not yield satisfying results: The applicability of the rules itselfs could not be sufficiently verified, and the main reason for undertaking it did return inconclusive results.
Even though the question if temporal connectors can be used in unsupervised learning of temporal relations could not be affirmed, it could also not be denied, and several possible remedies have been named to further explore this possibility.
The main insight that can be gained from this project is that temporal semantics and discourse semantics have different requirements in terms of what features are useful in each task, which might be surprising, considering that they are semantically related and that techniques from both of them can be applied to the other.

References

Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843. ACM ID: 358434.

Chambers, N. and Jurafsky, D. (2008). Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, page 698–706, Honolulu, Hawaii. Association for Computational Linguistics. ACM ID: 1613803.

Chambers, N. and Jurafsky, D. (2009). Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, page 602–610, Suntec, Singapore. Association for Computational Linguistics. ACM ID: 1690231.

Chambers, N., Wang, S., and Jurafsky, D. (2007). Classifying temporal relations between events. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, page 173–176, Prague, Czech Republic. Association for Computational Linguistics. ACM ID: 1557820.

Dorr, B. J. (1992a). A parameterized approach to integrating aspect with lexical-semantics for machine translation. In *Proceedings of the 30th annual meeting on Association for Computational Linguistics*, ACL '92, page 257–264, Newark, Delaware. Association for Computational Linguistics. ACM ID: 982000.

Dorr, B. J. (1992b). The use of lexical semantics in interlingual machine translation. *Machine Translation*, 7(3):135–193.

Dorr, B. J. and Gaasterland, T. (2002). Constraints on the generation of tense, aspect, and connecting words from temporal expressions. *Journal of AI Researchгео ртедс о тзжсй.*

Dorr, B. J. and Gaasterland, T. (2007). Exploiting aspectual features and connecting words for summarization-inspired temporal-relation extraction. *Information Processing & Management*, 43(6):1681–1704.

Dorr, B. J. and Olsen, M. B. (1997). Deriving verbal and compositional lexical aspect for NLP applications. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, EACL '97, page 151–158, Madrid, Spain. Association for Computational Linguistics. ACM ID: 979637.

Espinosa, J. and Lieberman, H. (2005). EventNet: inferring temporal relations between commonsense events. In Gelbukh, A., Albornoz, A., and Terashima-Marin, H., editors, *MICAI 2005: Advances in Artificial Intelligence*, volume 3789, page 61–69. Springer Berlin Heidelberg, Berlin, Heidelberg.

Gisborne, N. (2010). *The Event Structure of Perception Verbs*. Oxford University Press.

Lapata, M. and Lascarides, A. (2006). Learning sentence-internal temporal relations. *Journal of Artificial Intelligence Research*, 27(1):85–117.

Lascarides, A. and Asher, N. (1993). Temporal interpretation, discourse relations and commonsense entailment. *Linguistics and Philosophy*, 16(5):437–493.

Mani, I., Pustejovsky, J., and Gaizauskas, R. (2005). *The Language of Time: a reader*. Oxford University Press.

Mani, I., Verhagen, M., Wellner, B., Lee, C. M., and Pustejovsky, J. (2006). Machine learning of temporal relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, page 753–760, Stroudsburg, PA, USA. Association for Computational Linguistics. ACM ID: 1220270.

Mazur, P. and Dale, R. (2010). WikiWars: a new corpus for research on temporal expressions. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, page 913–922, Cambridge, Massachusetts. Association for Computational Linguistics. ACM ID: 1870747.

Moens, M. and Steedman, M. (1988). Temporal ontology and temporal reference. *Comput. Linguist.*, 14(2):15–28. ACM ID: 55058.

Olsen, M. B. (1994). The semantics and pragmatics of lexical aspect. *Studies in the linguistic sciences*, 24(2).

Puşcaşu, G. (2007). Wvali: Temporal relation identification by syntactico-semantic analysis. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, page 484–487.

Pustejovsky, J., Castano, J., Ingria, R., Saurí, R., Gaizauskas, R., Setzer, A., Katz, G., and Radev, D. (2003a). TimeML: robust specification of event and temporal expressions in text. In *IWCS-5 Fifth International Workshop on Computational Semantics*.

Pustejovsky, J., Hanks, P., Sauri, R., See, A., Gaizauskas, R., Setzer, A., Radev, D., Sundheim, B., Day, D., Ferro, L., et al. (2003b). The timebank corpus. In *Corpus Linguistics*, volume 2003, page 40.

Pustejovsky, J., Saurí, R., Setzer, A., Gaizauskas, R., and Ingria, B. (2002). TimeML annotation guidelines. *TERQAS Annotation Working Group*.

Reichenbach, H. (2005). The tenses of verbs. *The language of time: A reader*, page 71–78.

Setzer, A. and Gaizauskas, R. (2001). A pilot study on annotating temporal relations in text. In *Proceedings of the workshop on Temporal and spatial information processing*

- *Volume 13*, TASIP '01, page 11:1–11:8, Stroudsburg, PA, USA. Association for Computational Linguistics. ACM ID: 1118248.

Setzer, A., Gaizauskas, R., and Hepple, M. (2005). Using semantic inferences for temporal annotation comparison. In Mani, I., Pustejovsky, J., and Gaizauskas, R., editors, *The Language of Time: a reader*, page 575–584. Oxford University Press.

Steedman, M. (2010). Computational linguistic approaches to temporality.

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

A Tools used in the disambiguator

The following tools were used in the disambiguator:

CDTS-solution `ftp://ftp.umiacs.umd.edu/pub/bonnie/CDTS-Solution-2006.lsp`
A lisp script for splitting parsed input into relevant parts, extracting subject nodes, VP nodes, temporal connectors, main and adjunct clauses. Also provides a pattern matcher for finding associated tenses.

LCS lexicon `http://www.umiacs.umd.edu/~bonnie/verbs-English.lcs`
A lisp-readable lexicon of over 10,000 English verbs. It provides the following information for each verb:

- The lemmatised form of the verb
- The Levin class of the verb
- The WordNet sense key
- A PropBank style role-list string
- A grid of theta-roles
- An LCS representation
- Additional information about positions in the LCS form, e.g "optional", "animate +".

Stanford CoreNLP `http://nlp.stanford.edu/software/corenlp.shtml`
An NLP tool that provides a variety of functions useful for NLP tasks. It works on raw input and outputs an XML file.

PythonOnLisp `http://common-lisp.net/project/python-on-lisp/`
Pythononlisp provides an interface for accessing python modules in lisp, used here for lemmatising verbs and accessing WordNet.

NLTK `http://www.nltk.org/`
The Natural Language Toolkit, a python module that provides various APIs and functionality relevant for NLP tasks. Used here to access and process the various corpora, including WordNet, lemmatising,

B Disambiguation Rules

Require: before = {}, begins = {}, ends = {}, includes = {}, simultaneous = {}
  if connector ∈ {*after*, *before*} **then**
    before ∪ sentence
  else if connector == *once* **then**

```
    if matrix/adjunct ∪ {fut perf/pres,fut perf/pres perf} then
      ends ∪ sentence
    else if matrix/adjunct ∈ {fut/pres} then
      begins ∪ sentence
    else if matrix ∈ {progressive} or adjunct ∈ {progressive} then
      begins ∪ sentence
    else
      before ∪ sentence
    end if
  else if connector == since then
    if matrix/adjunct ∪ {simple/perfect} then
      begins ∪ sentence
    else if matrix/adjunct ∈ {pres/past} then
      before ∪ sentence
    else if matrix/adjunct ∈ {simple/simple} then
      if interval of adjunct == point then
        begins ∪ sentence
      else
        before ∪ sentence
      end if
    end if
  else if connector == as then
    if matrix/adjunct ∪ {fut/pres state} then
      ends ∪ sentence
    else if matrix/adjunct ∈ {fut/pres} then
      begins ∪ sentence
    else if matrix ∈ {progressive} then
      if interval of adjunct == point then
        includes ∪ sentence
      else
        simultaneous ∪ sentence
      end if
    else if adjunct ∈ {progressive} then
      if interval of matrix == point then
        includes ∪ sentence
      else
        simultaneous ∪ sentence
      end if
    else if matrix ∈ {perfect} and not adjunct ∈ {perfect} then
      begins ∪ sentence
    else if adjunct ∈ {perfect} and not matrix ∈ {perfect} then
      before ∪ sentence
    else if matrix/adjunct ∈ {simple/simple} then
      if interval of matrix == interval of adjunct then
        before ∪ sentence
      else if interval of matrix ∈ {open, closed} and interval of adjunct ∈ {open, closed}
      then
```

```
        includes ∪ sentence
      else
        begins ∪ sentence
      end if
    end if
  else if connector == while then
    if matrix ∈ {progressive} then
      if interval of adjunct == point then
        includes ∪ sentence
      else
        simultaneous ∪ sentence
      end if
    else if adjunct ∈ {progressive} then
      if interval of matrix == point then
        includes ∪ sentence
      else
        simultaneous ∪ sentence
      end if
    else if matrix == synonym(adjunct) or matrix == antonym(adjunct) then
      simultaneous ∪ sentence
    else
      includes ∪ sentence
    end if
  else if connector == when then
    if matrix ∈ {progressive} then
      if interval of adjunct == point then
        includes ∪ sentence
      else
        simultaneous ∪ sentence
      end if
    else if adjunct ∈ {progressive} then
      if interval of matrix == point then
        includes ∪ sentence
      else
        simultaneous ∪ sentence
      end if
    else if matrix/adjunct ∈ {fut/pres} then
      before ∪ sentence
    else if matrix/adjunct ∈ {simple/simple} then
      if np-sbj of matrix == np-sbj of adjunct then
        includes ∪ sentence
      else
        before ∪ sentence
      end if
    end if
  end if
```

C Lisp Code

```
(defun sort-into-timeml ()
  (let ((rel-table (make-array +relations-length+ :initial-element nil))
(once-f (make-zip (intersection *future* *perfect*) *present*)) ;ends
(once-s (make-zip (intersection *future* *simple-all*) (intersection *present* *simple*))) ;starts
(since-s (make-zip *perfect* *simple-all*)) ;starts
(since-b (make-zip *present* *past*)) ;before
(since-e (make-zip *simple* *simple-all*)) ;allowed tenses
(as-s (make-zip *future* (intersection *present* *simple*))) ;starts
(as-f (make-zip *future* (intersection *present* *simple-state*))) ;finishes
(when-b (make-zip *future* *present*)) ;before
filein
before inc sim begins ends)
    (with-open-file (in "when-before" :direction :input)
(setf filein (read in nil nil)) (setf before (append before filein)))
    (with-open-file (in "when-inc" :direction :input)
(setf filein (read in nil nil)) (setf inc (append inc filein)))
    (with-open-file (in "when-sim" :direction :input)
(setf filein (read in nil nil)) (setf sim (append sim filein)))
    (with-open-file (in "while-inc" :direction :input)
(setf filein (read in nil nil)) (setf inc (append inc filein)))
    (with-open-file (in "while-sim" :direction :input)
(setf filein (read in nil nil)) (setf sim (append sim filein)))
    (destructuring-bind (xx yy z) (array-dimensions *tense-table*)
      (dotimes (x xx)
(dotimes (y yy)
  ;after,before
  (setf before (append before (aref *tense-table* x y 0) (aref *tense-table* x y 2)))
  ;until
  (setf ends (append ends (aref *tense-table* x y 5)))
       ;once
       (cond ((member (list x y) once-f :test 'equal) (setf ends (append ends (aref *tense-table* x y 3))))
   ((member (list x y) once-s :test 'equal) (setf begins (append begins (aref *tense-table* x y 3))))
((or (member x *progressive*) (member y *progressive*)) (setf begins (aref *tense-table* x y 3)))
 (t (setf before (append before (aref *tense-table* x y 3)))))
  ;since
  (cond ((member (list x y) since-s :test 'equal) (setf begins (append begins (aref *tense-table* x y 4))))
   ((member (list x y) since-b :test 'equal) (setf before (append before (aref *tense-table* x y 4))))
((member (list x y) since-e :test 'equal)
   (and (setf begins (append begins
(intersection (aref *tense-table* x y 4)
      (aref *aspect-table* 3 1 4))
(intersection (aref *tense-table* x y 4)
      (aref *aspect-table* 3 2 4))
(intersection (aref *tense-table* x y 4)
      (aref *aspect-table* 3 3 4))
(intersection (aref *tense-table* x y 4)
      (aref *aspect-table* 1 3 4))
(intersection (aref *tense-table* x y 4)
      (aref *aspect-table* 2 3 4))
(intersection (aref *tense-table* x y 4)
      (aref *aspect-table* 3 3 4))))
(setf before (append before
(set-difference (aref *tense-table* x y 4)
(aref *aspect-table* 3 1 4))
(set-difference (aref *tense-table* x y 4)
(aref *aspect-table* 3 2 4))
(set-difference (aref *tense-table* x y 4)
```

```
(aref *aspect-table* 3 3 4))
(set-difference (aref *tense-table* x y 4)
(aref *aspect-table* 1 3 4))
(set-difference (aref *tense-table* x y 4)
(aref *aspect-table* 2 3 4))
(set-difference (aref *tense-table* x y 4)
(aref *aspect-table* 3 3 4)))))))
  ;as
  (cond ((member (list x y) as-s :test 'equal) (setf begins (append begins (aref *tense-table* x y 1))))
((member (list x y) as-f :test 'equal) (setf ends (append ends (aref *tense-table* x y 1))))
((member x *progressive*)
   (and (setf sim (append sim
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 1 3 1))
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 2 3 1))
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 3 3 1))))
(setf inc (append inc
(set-difference (aref *tense-table* x y 1)
(aref *aspect-table* 1 3 1))
(set-difference (aref *tense-table* x y 1)
(aref *aspect-table* 2 3 1))
(set-difference (aref *tense-table* x y 1)
(aref *aspect-table* 3 3 1))))))
((member y *progressive*)
   (and (setf sim (append sim
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 3 1 1))
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 3 2 1))
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 3 3 1))))
(setf inc (append inc
(set-difference (aref *tense-table* x y 1)
(aref *aspect-table* 3 1 1))
(set-difference (aref *tense-table* x y 1)
(aref *aspect-table* 3 2 1))
(set-difference (aref *tense-table* x y 1)
(aref *aspect-table* 3 3 1))))))
((and (member x *perfect*) (not (member y *perfect*)))
(setf begins (append begins (aref *tense-table* x y 1))))
((and (member y *perfect*) (not (member x *perfect*)))
(setf before (append before (aref *tense-table* x y 1))))
   (t
(and (setf sim (append sim
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 2 2 1))
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 2 1 1))
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 3 3 1))))
(setf begins (append begins
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 2 3 1))
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 3 2 1))
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 3 1 1))
```

```
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 1 3 1)))
(setf inc (append inc
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 1 1 1))
(intersection (aref *tense-table* x y 1)
      (aref *aspect-table* 1 2 1)))))))
  ;while
  (cond ((member x *progressive*)
 (and (setf inc (append inc
(intersection (aref *tense-table* x y 7)
      (aref *aspect-table* 1 3 7))
(intersection (aref *tense-table* x y 7)
      (aref *aspect-table* 2 3 7))
(intersection (aref *tense-table* x y 7)
      (aref *aspect-table* 3 3 7))))
(setf sim (append sim
(set-difference (aref *tense-table* x y 7)
(aref *aspect-table* 1 3 7))
(set-difference (aref *tense-table* x y 7)
(aref *aspect-table* 2 3 7))
(set-difference (aref *tense-table* x y 7)
(aref *aspect-table* 3 3 7))))))
((member y *progressive*)
   (and (setf sim (append sim
(intersection (aref *tense-table* x y 7)
      (aref *aspect-table* 3 1 7))
(intersection (aref *tense-table* x y 7)
      (aref *aspect-table* 3 2 7))
(intersection (aref *tense-table* x y 7)
      (aref *aspect-table* 3 3 7))))
(setf inc (append inc
(set-difference (aref *tense-table* x y 7)
(aref *aspect-table* 3 1 7))
(set-difference (aref *tense-table* x y 7)
(aref *aspect-table* 3 2 7))
(set-difference (aref *tense-table* x y 7)
(aref *aspect-table* 3 3 7)))))))
  ;when
       (cond ((member (list x y) when-b :test 'equal) (setf before (append before (aref *tense-table* x y 6))))
  ((member x *progressive*)
 (and (setf inc (append inc
(intersection (aref *tense-table* x y 6)
      (aref *aspect-table* 1 3 6))
(intersection (aref *tense-table* x y 6)
      (aref *aspect-table* 2 3 6))
(intersection (aref *tense-table* x y 6)
      (aref *aspect-table* 3 3 6))))
(setf sim (append sim
(set-difference (aref *tense-table* x y 6)
(aref *aspect-table* 1 3 6))
(set-difference (aref *tense-table* x y 6)
(aref *aspect-table* 2 3 6))
(set-difference (aref *tense-table* x y 6)
(aref *aspect-table* 3 3 6))))))
((member y *progressive*)
   (and (setf sim (append sim
(intersection (aref *tense-table* x y 6)
      (aref *aspect-table* 3 1 6))
```

```
(intersection (aref *tense-table* x y 6)
      (aref *aspect-table* 3 2 6))
(intersection (aref *tense-table* x y 6)
      (aref *aspect-table* 3 3 6))))
(setf inc (append inc
(set-difference (aref *tense-table* x y 6)
(aref *aspect-table* 3 1 6))
(set-difference (aref *tense-table* x y 6)
(aref *aspect-table* 3 2 6))
(set-difference (aref *tense-table* x y 6)
(aref *aspect-table* 3 3 6)))))))))
    (setf (aref *relations* 0) (delete-duplicates (append (aref *relations* 0) before)))
    (setf (aref *relations* 1) (delete-duplicates (append (aref *relations* 1) inc)))
    (setf (aref *relations* 2) (delete-duplicates (append (aref *relations* 2) sim)))
    (setf (aref *relations* 3) (delete-duplicates (append (aref *relations* 3) begins)))
    (setf (aref *relations* 4) (delete-duplicates (append (aref *relations* 4) ends)))
    (with-open-file
      (out "relations-final"
        :direction :output :if-exists :supersede :if-does-not-exist :create)
      (format out "~a" *relations*))))
```