Temporal Context Representation and Reasoning

Dan Moldovan, Christine Clark, Sanda Harabagiu
Language Computer Corporation
Richardson, Texas, 75080
{moldovan, christine, sanda}@languagecomputer.com

Abstract

This paper demonstrates how a model for temporal context reasoning can be implemented. The approach is to detect temporally related events in natural language text and convert the events into an enriched logical representation. Reasoning is provided by a first order logic theorem prover adapted to text. Results show that temporal context reasoning boosts the performance of a Question Answering system.

1 Introduction

Representing and reasoning about time-dependent information is important for applications ranging from databases, planning, natural language processing, and others. For example, temporal reasoning is essential in question answering to successfully addressing a time sensitive and dynamic world. Questions requiring temporal relations are not currently solvable in most state of the art question answering systems. Of the factoid TREC questions from 2001-2004 approximately 8.7% require temporal context reasoning. A temporal context scopes changing world views and serves as a tool for disambiguating time dependent events or intervals. Questions such as *Who was the president of South Korea when Bill Clinton was President of the US?* need temporal event detection and context based reasoning.

While there has been much theoretical research and modeling of context reasoning in the temporal domain, few systems have reported an evaluation of end to end temporal context enhanced question answering. This paper presents such a system. We report on (1) temporal context indexing for passage retrieval, (2) a fast and robust representation of time and annotation of time events, (3) the implementation of a machine learning based temporal event recognition module, (4) a model for a knowledge representation that maps natural language context into first order logic Suggested Upper Merged Ontology(SUMO)[Niles and Pease, 2001] predicates, and (5) a temporal context reasoning engine.

2 Approach

Several logic or semantic standard approaches to represent time (or events in time) have been defined. The most common ones include absolute dating schemes, relative event ordering or duration based approaches. Decisions about the nature of the event (instantaneous or extended) have to be made. The problem with many discrete schemes for natural language applications (such as Mani and Wilson, 2000) is the lack of absolute timestamps for many events. Propagation approaches to assign absolute time stamps to event clauses for sequence recognition have been implemented [Filatova and Hovy, 2001]. However, besides the recognition problems (Filatova and Hovy report 52% accuracy) these approaches do not capture temporal relations that are based on duration of events.

Relative event orderings and duration relations that hold true at all times should not be labeled with absolute timestamps as in: *People are born before they die*. This holds true also for event orderings commonly found in user manuals, like IT support information or cookbooks as in: *What process do I need to terminate after a failed query?*

[Allen, 1991] demonstrates a constraint propagation approach based on a set of possible interval relations that could capture this, but the recognition of these events and their relations is still an open research area.

Our approach uses a hybrid of the above mentioned methods. Passage retrieval makes use of a descrete time scale in the form of temporal indexing so as to capture all the relevant candidate passages that may not have absolute time specified in the document text. For the time-event model, high coverage signal words and phrases that specify how temporal elements should be related are used. The chosen signals are based on the annotations in TimeML used in the TERQAS project [Pustejovsky et al., 2002] and are mined for and annotated in a sampling of articles from the TREC collection. The signals are then classified by the type of temporal relation they express, such as overlapping, inclusion, ordering, and more. This data is used for training the automatic temporal event detection module that recognizes 7 distinct temporal event interval relations on the question and its answer candidates. The detected temporal events enrich a logical form representation of the question and answer candidates with

SUMO predicates and functions. Based on these temporal constraints, a context resolution engine operates as an answer candidate filter.

3 Passage Retrieval with Temporal Context

One common class of time dependent questions targets facts that are rooted in an absolute time, such as: *Who led the NHL in playoff goals in June 1998?* To ensure that all passages relevant to a temporally constrained question are retrieved it is necessary to index the temporal contexts in a document. The approach is to scan each document in a collection for its time stamp as well as any underspecified or relative references to time. A date resolution module processes all underspecified and relative dates to accurately anchor these temporal references in a calendar year. The context field consists of a *(year, month, day, hour, minute, second)* tuple, where any member in the tuple is optional. For a document with time, *T(1998, 06, 15, X, X, X)*, the following temporal context entries are computed and displayed in Table 1:

Temporal Reference	Resolved Date
last year	T(1999,X,X,X,X,X)
next week	T(1998,06,16-22,X,X,X)
in a week	T(1998,06,22,X,X,X)
today	T(1998,06,15,X,X,X)

Table 1: Resolved Temporal Context

The resolved references as well as the document time stamp is indexed and made searchable for time dependent queries. For the question above the query issued to the index is:
(_human) AND (lead) AND (nhl OR (national AND hockey AND league)3) AND (playoff) AND (goal) AND T(1998, 06, X, X,X,X,X)

For questions involving a time range, the query is translated into a conjunction of time operators. As an example, *What Chinese Dynasty was during 1412-1431?* translates to:
(_organization) AND (chinese) AND (dynasty) AND (T(1412,X,X,X,X,X) OR T(1413,X,X,X,X,X) OR ... OR T(1431,X,X,X,X,X))

Internally the search engine computes the allowed range and searches for dates indexed in that range. The temporal context for the question is enforced by searching the index for the context detected in the question and applying document dates to all passages retrieved that do not have an explicit context within a window of 3 sentences. These resolved dates are passed along to the answer processing module which invokes the signal based temporal event recognizer and the context resolution modules to verify that the candidate answers do meet the temporal constraints in the question

4 Data Annotation

For an initial analysis of the signal words, the TimeML [Pustejovsky et al., 2000] TimeBank4 data functioned as a source of time signal expressions to be used as seeding material in the TREC 2002 corpus. The signal list served as a means to bootstrap a sample of the TREC collection for annotation purposes. The goal of the annotation was to provide data for a machine learning algorithm aimed at disambiguating signal words, detecting time events, and determining the boundaries of the time event.

Category (example)	Interpretation	% Dist
Sequence (before, after)	E1 happened in full before E2	12.7
Containment (in, of)	E1 is contained by E2	30.2
Overlap (at,as,on)	An interval exists that is contained by E1 and E2	28.6
Right Open Interval (from, since)	E1 is the left boundary of E2, right is undefined	6.1
Left Open Interval (to, until)	E2 is the right boundary of E1, left is undefined	4.5
Closed Interval (for, all)	E1 lasts for the duration of E2	6.3
Absolute Ordering (first, last)	E1 has an ordering relative to E1	11.6

Table 2: Signal Annotation Categories and their Distribution

The requirements of the annotation scheme used to mark up 1000 sentences from NYT, 150 from APW, and 150 sentences from XIN were robustness, simplicity, and sufficient granularity of the time events.

Time events are defined as related intervals, which are bound by a signal expression indicating the relation of the time events. Additionally, each signal can modify up to two events and can determine one of the following relations: *sequence, containment, overlap, open interval (left or right), duration and absolute ordering.* An event is interpreted as an interval bound by signal words or phrases and can be noun phrases *(the bombing)* or verb groups *(participated)*.

Since the current model operates at the sentence level, a marker is added to signals that need more information to resolve its arguments such as implicit context or missing discourse. Table 2 lists the categories of signal words, their natural language interpretation, and their distribution across a sampling of the TREC collection.
Example annotations based on the signal categories:
Which country <E1 id=889> declared </E1 id=889> independence <S id=889 class=contain> in </S id=889><E2 id=889> 1776 </E2 id=889>?
<S id=358 class=sequence> After </S id=358> quickly <E1 id=358> decapitating </E1 id=358> the bird, Susan <E2 id=358> scalded </E2 id=358> the carcass.

5 Detection of Temporally Ordered Events

Automated discovery of temporally ordered events requires detecting a temporal triple **(S,E1,E2)** which consists of a time dependent signal word **(S)** and its corresponding temporal event arguments, **(E1)** and **(E2)**. Detection of a temporal triple is complicated by two issues:
(1) **Disambiguation of signal words:** Not all signal words are unambiguously classified as time indicators.
(a) *He stood* ***before*** *the judge* vs *He proofread the*

manuscripts ***before*** *mailing it to the publisher*
(b) *He woke up* ***at*** *10:00* vs *He looked through the window* ***at*** *the rising sun*
(2) **Attaching events to signal words**: Although some temporal events are found near their signal, many signals occur with their temporal events underspecified (non-local).
(a) *The problem was resolved* ***after*** *4 hours of intensive maintenance but* ***before*** *anybody was harmed.*

In the above example the signal word ***after*** easily attaches to *resolved* and *4 hours*, while the second signal word, ***before*** has a non-local E1 reference (also *resolved*).

To address these issues machine learning is employed in two stages, first to recognize and disambiguate signals, and second to discover and attach temporal events to their signals. The input to the learner is the data annotated. The predictive classifiers that result from learning are used to automatically detect temporal triples (signals with their attached temporal events) in natural language text.

Signal Disambiguation

In the first stage of the machine learning process, a signal's surface form is classified either as a temporal **(signal)** or non-temporal occurrence, with signals classified further as to how they order their events, see Table 2.

A chunk parse tree is examined for text segments that match the surface form of a signal which are placed in their own chunk if necessary. A set of features, listed in Table 3, is then computed and used to classify each of these chunks. The hyperterm feature can be described as a broadly applicable feature space constructed from SUMO. Each SUMO class, instance, or relation is a unique string, here called a term. All terms are collected and connected to their hyperterm via the SUMO axioms *(subclass) (subrelation)* and *(instance)*. The result is a directed acyclic graph connecting all 1020 terms to their hyperterms and rooted at the SUMO class *Entity*. This hyperterm tree along with the WordNet to SUMO mappings [Niles and Pease, 2003] is prepared offline. Using this resource to generate the hyperterm feature consists of computing the union of the hyperterms for all WordNet concepts (senses) for each word in the chunk.

The intuition behind using SUMO is: (1) Coarse grained distinctions appear to be better represented in SUMO hypterterm tree than in WordNet, that is, a single term "Process" appears to capture the idea of eventness easily where several WordNet hypernyms would be required to capture the same distinction. (2) All open class words are united in a single hyperterm tree whereas in WordNet verbs have shallow hypernymy and Adjective and Adverbs have no hypernymy. This allows for i) better generalization and similarity across chunks that have words not seen during training since unseen words will still share a significant set of hyperterm features and for ii) unified treatment of all four open lexical classes in the software. The resulting decision will be one of 8 choices corresponding to "No signal" and the 7 types of signals listed in Table 2.

Temporal Event Attachment

Once a signal has been identified, its associated events (E1 and E2) must be attached. This task is achieved with a local attachment algorithm followed by a signal chaining attachment algorithm.

Feature Description	Origin
Signal surface form	[Hindle and Rooth, 1993]
Chunk's phrase tag	[Kudo and Matsumoto, 2001]
Chunk's hyperterms	[new]
+/- 1 chunk's phrase tag	[Kudo and Matsumoto, 2001]
+/- 1 chunk's hyperterms	[new]
+/- 2 chunk's phrase tag	[Kudo and Matsumoto, 2001]
+/- 2 chunk's hyperterms	[new]

Table 3: Signal and Local Attachment Features and their Origin

The *Local Attachment Algorithm* proceeds from left to right evaluating each unattached chunk to its nearest signal by generating features for classification. The decision associated with the features is one of *Skip* (do not attach), *E1* (attach as E1), or *E2* (attach as E2). The features used and their origin are listed in Table 4.

The *Signal Chaining Attachment Algorithm* enables signals that are missing an attachment to pick up an attachment from another signal, called link signals. The algorithm proceeds left to right evaluating a signal's missing argument against another signal's existing argument. The decision is either *Skip*, *E1*, or *E2* wherein the latter two result in an attachment. The algorithm iterates until no attachments are made. Although this algorithm uses the features in Table 3, their contribution is low and so motivated the additional features listed in Table 4.

Feature Description	Origin
Missing signal argument (E1,E2)	[new]
Signal class	[mod. from [Allen, 1991]]
Link signal surface form	[Hindle and Rooth, 1993]
Link signal argument (E1,E2)	[new]
Link signal class	[mod. from [Allen, 1991]]
Parsing direction (left,right)	standard

Table 4: Signal Chaining Features and their Origin

Results

Each classifier was built using the LIBSVM [1] implementation of Support Vector Machines, with the radial basis kernel, $\exp(-\gamma\|u-v\|^2)$, and included performing a grid search to find the best model parameters. Measurements were taken by randomizing the annotated sentences and evaluating each tenth of the data by training on the other 90% (10-fold cross-validation). The results measure the performance of the module as a whole. Measuring individual decisions of the classifiers are not instructive due to the dominance of *NoSignal* decisions during disambiguation and *Skip* decisions during

[1] http://www.csie.ntu.edu.tw/ cjlin/libsvm/

attachment parsing.

Task	F-measure
Signal detection	71.4%
Attachment	49.1%

Table 5: Automated Detection Results

The attachment results are affected negatively by parsed chunk boundaries not matching the temporal event text segment marked by the annotator, which results in an annotated attachment being scored as incorrect. An attachment will also score as incorrect if it is not the correct argument of the signal, or if the signal was not classified correctly. No partial credit is given. In the current data set 41.8% of the attachment annotations fail to align with a chunk boundary. This indicates that temporal event boundaries should not always rely on the parse chunk boundaries. An attachment will also score as incorrect if it is not the correct argument of the signal, or if the signal was not classified correctly. The scoring program is strict in that a temporally ordered event is correct only when the signal has been classified correctly and both event arguments have been detected and assigned in the proper order.
Based on the results of our experiments reported in Table 5, with 1300 sentences from the TREC corpus, it seems apparent that more training data is needed. In order to verify that, the method and algorithm used for the current implementation is sound and scalable, another experiment was conducted using the same methodology over half the corpus (650 sentences). The results of that experiment posted an F-measure of 52.7% for signal detection and 36.1% for attachment. Comparing that to the original results it is clear that the technique outlined in this section scales well and will benefit from the ongoing effort to annotate more training data.

6 A Knowledge Representation with Temporal Context

The output of the Temporal Event Detection, described in section 4, is transferred to the SUMO enhanced logical form module. The function of this module is to translate the natural language candidate sentences, marked up with time event chunks, to a temporally enhanced first order logic assertion. The input time event chunks are labeled with the class of the signal from the list in Table 2, along with the parsed chunks identified as the time event arguments to the signal.

The Knowledge Representation for the temporally enhanced logic form is layered on top of the Logical Representation proposed in [Rus and Moldovan, 2002], where the first order logic structure is derived from the syntactic parse of the text and each token in the text is converted to a predicate. From this structure temporally related SUMO predicates are generated based on hand coded interpretation rules for the signal classes. The purpose of the interpretation rules is to define an algorithm for assigning a signal word to a SUMO predicate and defining the manner in which the slots for the predicate are determined. Table 6 enumerates the signal classes, and the SUMO predicate corresponding to the interpretation rule.

Signal Class	SUMO Logic
<S sequence, E1, E2>	earlier(E1,E2)
<S contain, E1, E2>	during(E1,E2)
<S overlap, E1, E2>	overlapsTemporally(E1,E2)
<S open_right, E1, E2>	meetsTemporally(E1,E2)
<S open_left, E1, E2>	meetsTemporally(E2,E1)
<S closed, E1, E2>	duration(E1,E2)

Table 6: Signal Class to SUMO Mapping

Once the SUMO predicate is chosen, the arguments to the predicate are the event argument ids from the heads of the chunks passed as attachments to the signal expression. Since all temporal SUMO predicates operate on time intervals, absolute times stated in the text are translated into a pair of time point predicates to specify the begin and end of the interval. A detailed example follows for the text: *That compares with about 531,000 arrests in 1993 before Operation Gatekeeper started.*

The temporal event recognizer disambiguates **in** and **before** as temporal signals, and classifies (1) **in** as a contain interval signal, and (2) **before** as a sequence interval signal. The *Local Attachment and Signal Chaining* algorithms then determine the time event arguments for each signal. The output triples are *(S1:contain=in, E1=arrests, E2=1993)* and *(S2:before=before, E1=arrests, E2=started)*. Once the mapping for the temporally ordered events and the absolute time events are complete, the SUMO predicates are generated. Predicates that were derived from signal words replace the signals in the logical form.
... 531_000_NN(x2) & arrest_NN(x2) & 1993_NN(x3) & Operation_NN(x4) & Gatekeeper_NN(x5) & nn_NNC(x6,x4,x5) & start_VB(e2,x6,x10) & ***earlier(WhenFn(x2), WhenFn(e2)) & Time(BeginFn(x11), 1993, 1, 1, 0, 0, 0) & Time(EndFn(x11), 1993, 12, 31, 23, 59, 59) & during(WhenFn(x3), x11) & during(WhenFn(x2), x3)***

7 Temporally Enhanced Inference

The temporal context resolution module employs a first order logic resolution style theorem prover adapted for natural language text processing [Moldovan et al, 2003]. The Set of Support (SOS) is the search strategy employed by the prover to partition the axioms into those that have support and those that are considered auxiliary [Wos, 1988]. This strategy restricts the search such that a new clause is inferred if and only if one of its parent clauses come from the SOS. The inference rule sets are based on hypperresolution, paramdodulation, and demodulation. Hyperresolution is an inference rule that does multiple steps in one, paramodulation introduces the notion of equality substitution for the demodulators required on the temporal axioms. Since temporal reasoning is intensive and may require long proofs, hyperresolution and paramodulation inference rules are necessary as they allow for a more compact and efficient proof by condensing multiple steps into one.

The inputs to the context resolution module include:
(1) The context relevant pieces of the natural language questions and candidate answers converted to a SUMO enhanced

first order logical form [Rus and Moldovan, 2002]. The question predicates are negated to invoke a proof by contradiction
(2) A set of empirically derived world knowledge axioms. Ex:
EVIDENCE IS SUPPORT FOR A CLAIM
evidence_NN(x2) -> support_NN(x2) & for_IN(x2,x3) & claim_NN(x3)
(3) Linguistic transformation rules derived from the input parse of the question and its candidate answers.
Ex: THE LEADER OF X, LEADS X.
_human_NE(x1) & leader_NN(x1) & of_IN(x1,x2) <-> lead_VB(e1,x1,x2)
(4) WordNet derived axioms such as lexical chains[Moldovan and Novischi, 2002] and glosses. Ex: TO START SOMETHING IS SEMANTICALLY RELATED TO ESTABLISHING SOMETHING
start_VB(e1,x1,x2) -> establish_VB(e1,x1,x2)
(5) A SUMO Knowledge Base of temporal reasoning axioms that consists of axioms for a representation of time points and time intervals, Allen primitives, and temporal functions.
Ex: DURING IS A TRANSITIVE ALLEN PRIMITIVE
during(TIME1, TIME2) & during(TIME2, TIME3) -> during(TIME1, TIME3)

Axioms in set (1) are loaded into the Set of Support. The predicates representing the temporally relevant information in the question are *anded* together, negated, and universally quantified, so as to invoke a proof by contradiction. The temporal predicates for the candidate answer are also *anded* together, existentially quantified and loaded into the SOS. The axioms in sets (2) - (3) are loaded into the Usable list to assist in the inference process between the question and the answer.

The output of the temporal context module comprises filtered and re-ranked answers that are scored based on their semantic and temporal similarity to the question. The inference engine will continue seeking a proof until any of the *TIME* predicates are unsatisfied. Partial satisfiability is permitted for other predicates in the question via a backoff algorithm integrated in the inference engine [Moldovan et al, 2003].

The following example illustrates how a temporal context resolution module is used to verify the temporal constraints of a question over a candidate answer:

Question: *What country controlled Syria in 1930?*
Question Logical Form:
_country_AT(x2) & control_VB(e1,x2,x1) & Syria_NN(x1) & overlapsTemporally_CTMP(e1,x3) & Time_CTMP(BeginFn(x3),1930,1,1,0,0,0) & Time_CTMP(EndFn(x3),1930,12,31,23,59,59)
Question Axiom:
-(exists e1 x1 x2 x3 (_country_AT(x2) & control_VB(e1,x2,x1) & Syria_NN(x1) & overlapsTemporally_CTMP(e1,x3) & Time_CTMP(BeginFn(x3),1930,1,1,0,0,0) & Time_CTMP(EndFn(x3),1930,12,31,23,59,59)))
Candidate Answer 5:
France has a different relationship with Syria, partly because it was once a French-mandated territory, from 1920-1946
Answer Logical Form:
France_NN(x1) & _country_NE(x1) & have_VB(e1,x1,x2) & different_JJ(x2) & relationship_NN(x2) & with_IN(x2,x6) & Syria_NN(x6) & partly_RB(e1) & because_IN(e1,e3) & once_RB(e3) & French_NN(x5) & mandate_VB(e3,x5,x6) & territory_NN(x6) & during_CTMP(x6,x9) & Time_CTMP(BeginFn(x9),1920,1,1,0,0,0) & Time_CTMP(EndFn(x9),1946,12,31,23,59,59)
Answer Axiom:
exists e1 e2 x1 x2 x3 x4 x6 x7 x8 x9 (France_NN(x1) & _country_NE(x1) & have_VB(e1,x1,x2) & different_JJ(x2) & relationship_NN(x2) & with_IN(x2,x6) & Syria_NN(x6) & partly_RB(e1) & because_IN(e1,e3) & once_RB(e3) & French_NN(x5) & mandate_VB(e3,x5,x6) & territory_NN(x6) & during_CTMP(x6,x9) & Time_CTMP(BeginFn(x9),1920,1,1,0,0,0) & Time_CTMP(EndFn(x9),1946,12,31,23,59,59))
Linguistic Axiom:
all x1 (French_NN(x1) → France_NN(x1))
Lexical Chain Axiom:
all e1 x1 x2 (mandate_VB(e1,x1,x2) → control_VB(e1,x1,x2))
SUMO Axioms:
Two time intervals overlapsTemporally if and only if each one is a temporalPart of the other:
all I1 I2 (ISA(I1, TimeInterval) & ISA(I2, TimeInterval) & overlapsTemporally_CTMP(I1, I2) ↔ (exists I3 (ISA(I3, TimeInterval) & temporalPart_CTMP(I3, I1) & temporalPart_CTMP(I3, I2))))
Interval1 is temporalPart of Interval2 if Interval1 is during Interval2:
all T1 T2 (ISA(T1, TimeInterval) & ISA(T2, TimeInterval) & during_CTMP(T1, T2) → temporalPart_CTMP(T1, T2)).
Interval1 is during Interval2 if Interval1 starts after Interval2 and Interval1 ends before Interval2:
all I1 I2 ((before_CTMP(EndFn(I1), EndFn(I2)) & before_CTMP(BeginFn(I2), BeginFn(I1))) → during_CTMP(I1,I2)).
A mapping of question predicates to the inferred answer predicates is provided in Table 7:

Question Predicates	Answer Predicates
_country_AT(x2)	French_NN(x5) → France_NN(x5)
control_VB(e1,x2,x1)	mandate_VB(e3,x5,x6)
Syria_NN(x1)	Syria_NN(x6)
overlapsTemporally_CTMP(e1, x3) & Time_CTMP(BeginFn(x3), 1930, 1, 1, 0, 0, 0) & Time_CTMP(EndFn(x3), 1930, 12, 31, 23, 59, 59)	during_CTMP(x6, x9) & Time_CTMP(BeginFn(x9), 1920, 1, 1, 0, 0, 0) & Time_CTMP(EndFn(x9), 1946, 12, 31, 23, 59, 59)

Table 7: Question to Answer Predicate Resolution

Proofs for answers which do not have the correct context will fail, resulting in the candidate answer being pruned from the answer list. This in turn promotes all candidates below. As an example, the context resolution module will fail to find a proof for the first candidate answer returned by the answer processing module (without temporal reasoning), whose document date is 1998:
The United States did not punish Israel when it occupied territories of some Arab countries such as Palestine and Syria, and refused to comply with relevant U.N.resolutions on the Middle East issues.

Since the document date of the candidate does not meet the temporal constraints of the question, the incorrect answer is filtered from the list and correct answer, candidate 5 is rightly promoted to position 1.

8 Question Answering Results

A set of 200 time dependent questions was used as a benchmark for measuring the contribution of temporal context to a state of the art question answering system. The benchmark was created primarily of questions that required temporal reasoning and so were previously unanswerable. Further, they were derived from a 300 MB sub-collection of the TREC corpus that included a sample of XIN, APW, and NYT articles. Table 8 summarizes the results of the system at the exact and sentence answer level with and without temporal context. The first two columns list QA results for the percent of correct answers in the first position and the Mean Reciprocal Rank, without temporal context. The second two columns list the corresponding results for temporally enhanced QA. The final column gives the percent increase for answers in the first position.

answer type	no context		context		increase in first
	first	MRR	first	MRR	
exact	16.4%	.191	20.0%	.259	22.0%
sentence	29.1%	.328	37.1%	.381	30.0%

Table 8: QA Results

9 Discussion

A means to represent and reason about context is crucial for the performance of a question answering system over a time sensitive corpus. Advantages include: retrieving answer passages with relative or underspecified dates, discarding temporally inaccurate answers, and capturing the ordering of events via signal words. Temporal inference is expensive, however, and should be ameliorated by development of a special purpose reasoner. Further, the performance of signal and event detection should be improved by the ongoing effort to annotate training data, as demonstrated by the scaling experiments. Despite this, the results presented in this paper have shown that significant improvement in open domain question answering can in fact be achieved by integrating temporal context reasoning.

10 Acknowledgements

This work was supported in part by the ARDA AQUAINT program. Special thanks to Bob Hauser, Arthur Dexter, and Jens Stephan for their invaluable research, implementation, and annotation efforts.

References

[Allen, 1991] James Allen. *Time and Time Again: The Many Ways to Represent Time*. International Journal of intelligent Systems, July 1991, pp341-355.

[Filatova and Hovy, 2001] Elena Filatova and Eduard Hovy. 2001. *Assigning Time-Stamps to Event-Clauses*. Proceedings of ACL Workshop on Temporal and Spatial Reasoning. Toulouse, France.

[Hindle and Rooth, 1993] Donald Hindle and Mats Rooth 1993 *Structural ambiguity and lexical relations*. Computational Linguistics, 19(2):313-330

[Kudo and Matsumoto, 2001] Taku Kudo and Yuji Matsumoto 2001 *Chunking with Support Vector Machines.* Proceedings of the NAACL, pages 192-199.

[Mani and Wilson, 2000] Inderjeet Mani and George Wilson. 2000. *Robust Temporal Processing of News*. Proceedings of the 38th Annual Meeting of the ACL. Honkong, China

[Moldovan and Novischi, 2002] D. Moldovan and A. Novischi 2002 *Lexical Chains for Question Answering.* Proceedings of COLING 2002, pages 674-680.

[Moldovan et al, 2003] D. Moldovan, C. Clark, S. Harabagiu, S. Maiorano 2003 *COGEX: A Logic Prover for Question Answering.* Proceedings of the Human Language Technology and North American Chapter of the Association for Computational Linguistics 2003, pages 87-93.

[Niles and Pease, 2001] Ian Niles and Adam Pease. 2001. *Towards a Standard Upper Ontology*. Proceedings of the 2nd International Conference on Formal Ontology in Information Systems. Ogunquit, Maine, October 2001

[Niles and Pease, 2003] Ian Niles and Adam Pease. 2003. *Linking Lexicons and Ontologies: Mapping WordNet to the Suggested Upper Merged Ontology*. Proceedings of the 2003 International Conference on Information and Knowledge Engineering (IKE '03). Las Vegas, Nevada, June 23-26, 2003

[Pustejovsky et al., 2000] Pustejovsky, Castano, Ingria, Sauri, Gaizauskas, Setzer, Katz. 2000. *TimeML: Robust Specification of Event and Temporal Expressions in Text*. Fifth International Workshop on Computational Semantics

[Pustejovsky et al., 2002] James Pustejovsky. 2002 *TERQAS: Time and Event Recognition for Question Answering Systems*. An ARDA Workshop on Advanced Question Answering Technology, January-July 2002.

[Rus and Moldovan, 2002] V Rus and D. Moldovan. 2002 *A High Performance Logic Form Transformation.* International Journal on Artificial Intelligence Tools, Vol. 11, No. 3 (2002) 437-454

[Wos, 1988] Larry Wos. 1998. *Automated Reasoning, 33 Basic Research Problems*. Prentice Hall, 1988.