Interpreting the Temporal Aspects of Language

by

Naushad UzZaman

Submitted in Partial Fulfillment

of the

Requirements for the Degree

Doctor of Philosophy

Supervised by

James F. Allen

Department of Computer Science
Arts, Science & Engineering
Edmund A. Hajim School of Engineering & Applied Sciences

University of Rochester
Rochester, New York

2012

To my family:

my loving parents, Mohammad Akhtaruzzaman and Marzina Mahfuz,

my sweet wife Naima Elahi,

my caring brother Arshad Uzzaman and his wife Fhamida Khatun.

It is because of your love, support and inspiration I came this far.

Curriculum Vitae

Naushad was born in Dhaka, Bangladesh on January 26th, 1983. He began his undergraduate studies at BRAC University, Bangladesh in 2001, graduating in 2006 with a Bachelor of Science degree in Computer Science. From 2005 to 2007, he worked as a Research Programmer under the supervision of Professor Mumit Khan in the Center for Research on Bangla Language Processing (CR-BLP) at BRAC University. In the Fall of 2007, Naushad started his graduate studies in Computer Science at the University of Rochester, Rochester, NY. He specialized in Computational Linguistics (Natural Language Processing) – a branch of Artificial Intelligence, under the direction of Professor James F. Allen. He received his Master of Science degree in Computer Science from the University of Rochester in 2009. He has done multiple research internships in various domains, such as, game prediction using social media (Yahoo! Research – Summer 2011), medical NLP (Microsoft Medical Media Lab – Summer 2009), and car dialog systems (Bosch Research and Technology Center – Summer 2008).

Acknowledgments

First, I want to sincerely thank my advisor James F. Allen for always being very welcoming, considerate, understanding, and encouraging. He gave me the freedom to pursue my own research, and at the same time provided me with ideas and support whenever I needed – not to mention the torture I did to him with the first draft of my documents, including this thesis, which he patiently went through and returned with insightful feedback.

My special thanks to Jeffery P. Bigham for giving me the opportunity to be exposed to the Human-Computer Interaction (HCI) area, and to collaborate on exciting projects. I want to especially thank Lenhart K. Schubert for giving me constructive suggestions on various topics and for our fruitful discussions on the Timegraph (section 5.3.1). I am genuinely indebted to my committee members – James F. Allen, Jeffrey P. Bigham, Lenhart K. Schubert, Daniel Gildea and Greg Carlson – for their constant feedback and suggestions every six months.

I also want to thank Hector Llorens, whose visit to Rochester was very productive for both of us. I greatly enjoyed collaborating with him. We jointly worked on merging temporal annotations (section 7.2), using temporal question answering for temporal evaluation (section 6.2), and organizing TempEval-3 (section 7.3). I would also like to thank my other co-organizers of TempEval-3 – James Pustejovsky, Marc Verhagen and Leon Derczynski for their support, contributions and feedback. My heartiest thanks to Hector and Leon for their critical suggestions on the first draft of the thesis as well.

Benjamin Van Durme played the role of my *big brother* in the academia. He always gave me helpful advice and new ideas every time I talked to him. Also thanks to Mary Swift, William DeBeaumont and George Ferguson for their help with the TRIPS parser.

All of my internships were very rewarding in terms of learning and exploring new domains and problems. I would like to sincerely thank all my mentors from the internships - Jungling Hu from Bosch RTS; Hank Rappaport, PD Singh and Michael Gillam from Microsoft Medical Media Lab; and Michael Matthews, Roi Blanco and Peter Mika from Yahoo! Research.

As graduate school can be a very frustrating place, having supportive friends is always helpful. I want to thank all of my Bangladeshi friends in Rochester, my friends from the *Fall 2007 class* – especially for their support in our first year, my officemates, my photography buddies and the very helpful department staff. JoMarie Carpenter, Martha Guenther and Iftekhar Naim especially deserve to have their names mentioned, but all of you made my stay at Rochester very pleasant and memorable.

My very special acknowledgement to my undergraduate advisor from BRAC University (Bangladesh), Professor Mumit Khan, who literally shaped my career. I was really fortunate to have him as my mentor at the right time of my career. I learned a lot from him and I am very grateful to him.

My parents, Mohammad Akhtaruzzaman and Marzina Mahfuz, my brother Arshad Uzzaman, and his wife Fhamida Khatun, have supported me in every step. My father always guided me and my mother constantly inspired me to do better. Without the unconditional love of my family I could not have come this far.

Last but not the least, my lovely wife, Naima Elahi (Neema), has been with me since the beginning of my graduate school, whose love has been a constant support for my survival in this significant journey of my career. I would, also, like to wholeheartedly thank her for thoroughly proofreading the thesis, which clarified my thesis significantly.

Abstract

Understanding temporal information in natural language text is fundamental for deep language understanding, and key to many advanced natural language processing (NLP) applications, such as question answering, information extraction, document summarization and dialog systems. These techniques can be applied in news, medical, history and other domains.

In this dissertation, we first present our hybrid system to automatically extract temporal information from raw text by extracting events, temporal expressions and identifying temporal relations between entities. Our system had a competitive performance in the temporal evaluation shared task - TempEval 2010. Then we present a metric that we developed for the evaluation of temporal annotation. Our metric has been adopted by the premier temporal evaluation shared task, TempEval 2013, to evaluate participating systems. We also present a question-answering (QA) system that can answer temporal questions with temporal reasoning. Our developed QA system can be used to evaluate temporal information understanding capability. Finally, we describe our contributions in improving the existing temporal resources.

Contributors and Funding Sources

This work was supervised by a dissertation committee consisting of Professor James F. Allen (advisor), Professor Lenhart K. Schubert, Professor Jeffery P. Bigham and Professor Daniel Gildea of Home Department and Professor Greg Carlson of Outside Department. The work on merging temporal annotations (section 7.2), using temporal question answering for temporal evaluation (section 6.2), and organizing TempEval-3 (section 7.3), was conducted in part by Hector Llorens. All other work conducted for the dissertation was completed by the student independently, with supervision from his advisor.

This material is based upon work supported by ONR #'s N00014-11-1-0417, N00014-01-1-1015, NSF IIS-1012017 and NIH R01-AG033202-3. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of above named organizations.

Table of Contents

List of Tables

List of Figures

1 Introduction

Understanding temporal information in natural language text is required for deep language understanding. Researchers from different areas, such as linguistics, computer science, philosophy, and cognitive science, have been working for the last few decades on understanding the temporal information in language; but it has not been of much interest to the larger Artificial Intelligence (AI) community until very recently. The recent emergence of Natural Language Processing (NLP) applications, such as, question answering, summarization, information extraction and dialog systems have drawn attention to the need for temporally aware systems, so that these applications can extract temporal meanings from documents. Such capabilities will enable us to implement better NLP applications.

For example, a doctor wants to know from a patient's historical records if the patient had any abdominal pain a few months before the CT scan was performed. To answer this question automatically from the patient's historical records, we need a system with temporal information understanding capability, which would help us to *automatically extract the medical events, the temporal expressions, classify the temporal relations,* and then *make inferences to identify how these events are related to each other in terms of time.*

Moreover, understanding temporal information would help in many other domains and applications as well. For example, in news domains, having an automated system to summarize articles in chronological order or a system for temporal visualization could help many readers, e.g. individuals who have trouble reading and understanding, be it dyslexic people or those who are not native English speakers. Temporal information extraction would benefit in almost any applications involving the processing of natural language text.

1.1 Temporal Information in Natural Language

In this section we demonstrate a toy example to explain why temporal information understanding is hard and why we need to use ideas from many different areas, such as, linguistics, symbolic artificial intelligence, corpus linguistics, etc. to build an automated system to understand temporal information. Consider the example in Figure 1.1.

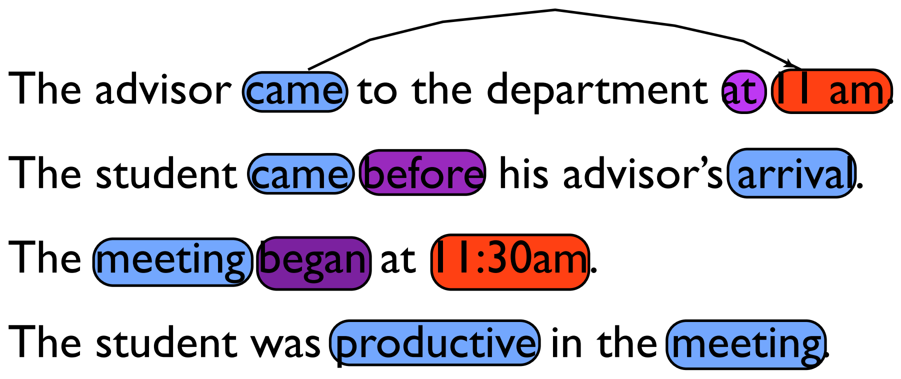
The advisor came to the department at 11 am.
The student came before his advisor's arrival.
The meeting began at 11:30am.
The student was productive in the meeting.

Figure 1.1: A toy example

To understand the temporal information from this text, the first task is to extract the temporal entities: **events**[1] – *came*$_{advisor}$, *came*$_{student}$, *arrival, meeting, productive* and *meeting*; and **temporal expressions** – *11am* and *11:30am*. Next, we have to **identify how the temporal entities are related** to each other: the first pair of entities, *came*$_{advisor}$ and 11*am*, is *simultaneous* with each other, but the next pair, *came*$_{student}$ and *arrival*, is not *simultaneous*, rather the student *came before* the advisor's *arrival*. Here we have the same word, *came*, in both cases, but because of the context and signals, such as *at* and *before* the temporal relations are different.

After extracting and interpreting this temporal information, we still have implicit information that is not explicitly mentioned in the document. For example, human readers can infer that the student came before the meeting, as the student came before the advisor's arrival, which happened before the meeting started. However, this information is not explicitly mentioned and we need to build a mechanism, such as a temporal structure (shown in Figure 1.2) to show how all entities are related to each other in terms of time. With such mechanism, we can answer temporal questions from documents by doing the temporal reasoning.

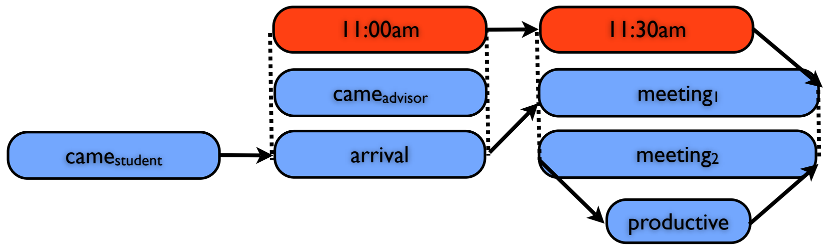

Figure 1.2: Temporal structure for the toy example in Figure 1.1

In the next section, we list the research questions that we answer in this thesis.

[1] *Events* are loosely referred to *events* and *states*.

1.2 Research Questions

We attempt to answer the following research questions in this thesis:

1. How can we build and improve the state-of-the-art in temporal information extraction and temporal relation identification?
2. How can we answer temporal questions with temporal reasoning?
3. How can we improve the existing temporal evaluation metrics?
4. How can we improve the existing temporal resources?

1.3 Contributions

In this thesis, we develop techniques to solve a portion of the language understanding problem – temporal information understanding. Our techniques of combining deep language understanding and machine learning classifier attain the state-of-the-art performance in temporal information extraction tasks. Next we present a temporal question-answering system capable of temporal reasoning. Our system can answer *yes/no*, *list* and *factoid* questions. We propose new temporal evaluation metrics to evaluate the automated systems extracting temporal information from text. We also improve the existing temporal resources to advance the research in this area.

In the next section, we describe how we structure the thesis.

1.4 Thesis Structure

Recent work on Computational Linguistics or Natural Language Processing (NLP) community is mostly driven by corpus based approaches. Our implemented system is also influenced by the same approaches. Additionally, we also borrow many ideas from Artificial Intelligence and Linguistics research to have a deeper understanding of temporal information. In **Chapter 2**, we describe the related work on analytical linguistics, philosophy and symbolic AI, and in **Chapter 3**, we describe the corpus linguistics based approaches for the task. **Chapter 2** and **3** include the temporal information understanding background and the summary of current approaches.

Then in **Chapter 4**, we describe our system ***TRIOS*** and ***TRIPS*** for understanding temporal information from texts. Our approach for all the tasks is best described as a hybrid between linguistically motivated solutions and machine learning classifiers, in which we try to get the best from both worlds. Next in **Chapter 5** we implement a temporal question-answering system by doing temporal reasoning.

In **Chapter 6**, we propose new evaluation metrics to evaluate temporal information from text. Our proposed metric has been adopted for evaluating the next Temporal Evaluation Shared Task. Based on temporal question answering, we develop another evaluation metric that can evaluate the temporal information understanding capability.

Next in **Chapter 7**, we describe the contributions of this thesis in improving the existing temporal resources. Finally in **Chapter 8**, we summarize our contributions and discuss possible future directions of this research work.

2 Related Linguistics and Artificial Intelligence Research

In this chapter, we explore the existing research done on Linguistics and Artificial Intelligence areas related to temporal information understanding. We borrow many insightful ideas from this research in the thesis.

We start the chapter by introducing linguistic theories on *tense* and *lexical aspect* to understand *time* in language. Next, we discuss the computational approaches from Artificial Intelligence to *represent* and *reason* about *time*.

2.1 Tense

In order to draw conclusions about time-related events, we have to understand the temporal information conveyed by the language. To understand the temporal location and relations of events, the first thing we should consider is *tense*. Tense usually gives us information about when the event occurred, i.e. is it before the speech time, or during the speech time, or after the speech time, etc. Consider the following examples:

(2.1). I **knew** the *world is flat.* (past tense)

(2.2). I **know** the *world is not flat.* (present tense)

(2.3). I **will know** the *world is round.* (future tense)

In English, the present tense usually locates events as occurring roughly at the speech time; past tense usually refers to a time prior to the speech time; and future tense usually refers to a time after the speech time. The examples (2.1), (2.2) and (2.3) above follow these rules and we can easily identify that knowing *world is flat* occurred before knowing *world is not flat* and knowing that the *world is round* occurred after knowing that the *world is not flat.*

However, the relation between simple verb tense and points in time is not straightforward. For example, the use of present tense does not always mean that the event coincides with the speech point. The present tense can also be used to locate events as occurring in the past, as in *"then he tells me..."*, or in the future, as in *"She leaves next week"*, and also to express habitual events, as in *"he loves movies"*.

More complications occur when we consider some of the other verb tenses. Consider the following examples:

(2.4). He arrived late.

(2.5). He had arrived late.

Although both refer to events in the past, representing them in the same way seems incorrect. Example (2.5) seems to refer to another unnamed event, i.e. *he arrived late* in reference of some other event. The generally accepted solution to handle this phenomenon is proposed by Reichenbach (1947) (Rei47), who suggests the notion of **reference time**. Reichenbach develops a theory that *tense* gives information about three types of *times*:

S - the time of speech

R - the reference time

E - the time of event/state

The reference time can be provided by temporal adverbs, or often can be determined by the discourse context. Here we do not present Reichenbach's original SRE relations, but instead discuss the variant described as **SRE triples** by Song and Cohen (1991) (SC91). Song and Cohen's list covers the same number of tenses as Reichenbach's, but unlike Reichenbach's their list is unambiguous and precise. In presenting these SRE triples, we keep the names of these tenses as proposed by Reichenbach.

In the simple tenses the reference time is same as the event time, i.e. E = R. The three forms are generated by varying the relationship between R and S:

Simple Past	Simple Present	Simple Future
(S > R = E)	(S = R = E)	(S < R = E)
Naima sang	Naima sings	Naima will sing

The perfect tenses, on the other hand, have the event time preceding the reference time, and differ in terms of how the reference time and speech time are related. Thus we have:

Past Perfect	Present Perfect	Future Perfect
(S > R > E)	(S = R > E)	(S < R > E)
Naima had sung	Naima has sung	Naima will have sung

This analysis also provides an account of the posterior tenses, in which R < E:

Posterior Past	Posterior Present	Posterior Future
(S > R < E)	(S = R < E)	(S < R < E)
Naima was going to sing	Naima is going to sing	Naima will be going to sing

In the current corpus-based approaches, *tense* information is widely used for temporal relation classification. Most of the systems combine the *tense* and *aspect*[1] attributes (section 3.1.1) to get similar benefits of Reichenbach's distinctions with the machine learning classifiers. Influenced by Reichenbach, Derczynski and Gaizauskas (2011) (DG11) also propose a markup language for the tenses of verbs and temporal relations between verbs.

2.2 Lexical Aspect

In this section we explore the temporal properties of *events*, such as whether events last, change, or complete. This would help us to identify more fine grained relationships (described in section 2.6) between events.

Lexical Aspect or **Aktionsarten** (German for *kind of action*) distinguishes between different subclasses of events based on the temporal properties, such as whether an event has ended or is ongoing, whether it is happening at a point in time, or over a period of time. In the literature on *aspect*, Vendler's (1967) (Ven67) work has been the foundation for future research. Considering the properties described above, Vendler divides the event expressions into four general classes: **states (stative)**, **processes (activity)**, **accomplishments**, and **achievements**. Following are the descriptions of each of the event expressions, as well as some of the diagnostic techniques suggested in Dowty (1986) (Dow86) for identifying examples of each kind.

[1]This *aspect* is different from *lexical aspect*. Check section 3.1.1 for details.

An event is a **stative** if it has the *subinterval property*, i.e. if its truth at an interval implies that it is true at all subintervals. Example (2.6) and (2.7) are example of stative.

(2.6). Sharif is happy.

(2.7). I believe the world is flat.

There are a number of ways to identify statives. For example, stative verbs are distinctly odd when used in the progressive form.

(2.8). *Sharif is being happy.

(2.9). *I am believing that the world is flat.

Statives are also odd when used as imperatives.

(2.10). *Believe that the world is flat.

An event is an **activity** if it has a *restricted subinterval property*, i.e. its truth at an interval implies that it is true for most subintervals. Example (2.11) to (2.13) are instances of activity expressions.

(2.11). Puspa is running.

(2.12). Neema lives in New York.

(2.13). Arshad drove a BMW.

Unlike statives, activity expressions are fine in both the progressive and imperative forms.

(2.14). Neema is living in New York.

(2.15). Drive a BMW!

However, like statives, activity expressions do not allow temporal modifiers, such as *in five minutes* (example (2.16) to (2.18)), but they allow duration modifiers, such as *for five minutes* (example (2.19) to (2.21)).

(2.16). *Neema lives in New York in a month.

(2.17). *Jhuma drove a Mini in an hour.

(2.18). *Marzina is running in an hour.

(2.19). Neema lived in New York for a month.

(2.20). Jhuma drove a Mini for an hour.

(2.21). Marzina is running for an hour.

Accomplishment expressions, on the other hand, describe events that have a natural end point and result in a particular state. Example (2.22) and (2.23) are instances of accomplishment.

(2.22). They climbed the mountain in two days.

(2.23). Marzina fell asleep in an hour.

In these examples, there is an event that is seen as occurring over a period of time that ends when the intended state is accomplished.

Activities and accomplishments can be distinguished by how they are modified by distinct temporal adverbials. In general, accomplishments can be modified by "in" temporal expressions, while simple activities cannot. See example (2.24) and (2.25).

(2.24). *Neema lived in New York in a year.

(2.25). *Marzina stayed at home in a month.

Finally, similar to accomplishment expressions, the **achievement** ones result in states. A few examples of achievement expressions are below.

(2.26). Neema recognized the man.

(2.27). Marzina woke up.

However unlike accomplishments, an achievement event happens in an instant and is not equated with any particular activity leading up to the state. Moreover, unlike activity and accomplishment expressions, achievements cannot be modified by in/for adverbials.

Both accomplishments and achievements are events that result in states. They are sometimes characterized as subtypes of a single aspectual class called **telic eventualities** (culminates). Both types of events have the property that if they are true at an interval, they are false at any subintervals of the intervals.

Summary of the properties of aspectual classes are given in Table 2.1 and these classes are shown pictorially in Figure 2.1 (due to (Pas88)).

Aspectual class	Dynamic	Culminate	Durative	Example
Stative	NO	NO	YES	*know, have*
Activity	YES	NO	YES	*walk, paint*
Accomplishment	YES	YES	YES	*build, destroy*
Achievement	YES	YES	NO	*notice, win*

Table 2.1: Different properties of the aspectual classes

Moens and Steedman (1988) (MS88) extend Vendler's classification on event ontology. They present a *tripartite event structure*, which captures if an event is in *preparatory phase*, *culmination* or in *consequent phase*. This

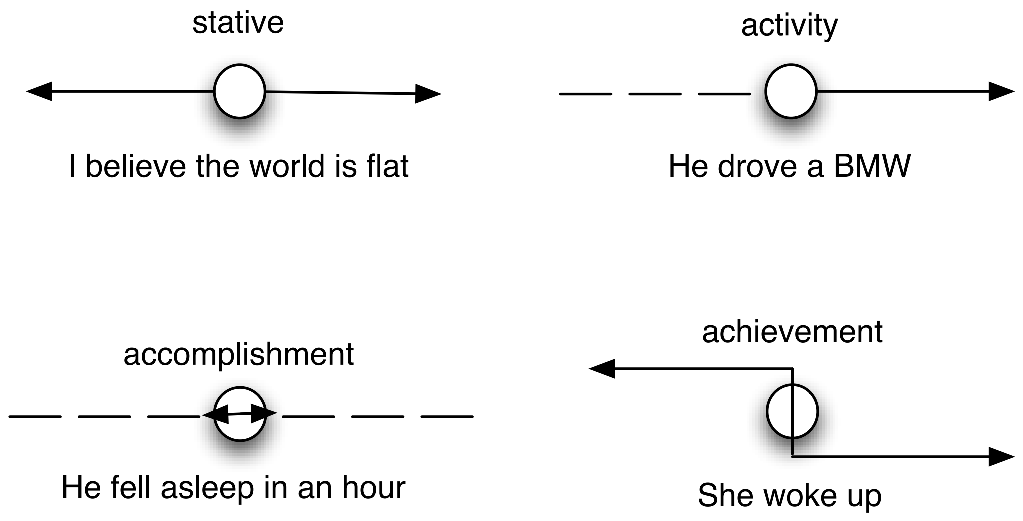

Figure 2.1: Pictorial Representations of Lexical Aspect

event ontology is shown pictorially in Figure 2.2 and with an example in Figure 2.3.

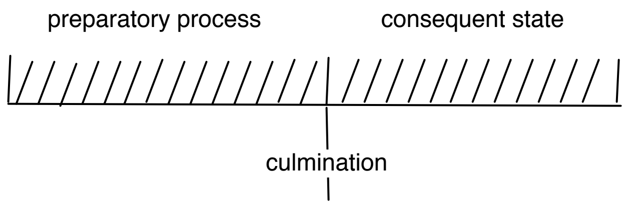

Figure 2.2: Tripartite event ontology by Moens and Steedman (MS88)

They introduce another class of events called **points** which are instantaneous and involve no culmination. Additionally, their ontology includes a computational model for the aspect calculus of events. The properties of aspectual categories and the lexical aspect shifts among these properties are modeled in a transition network (Figure 2.4).

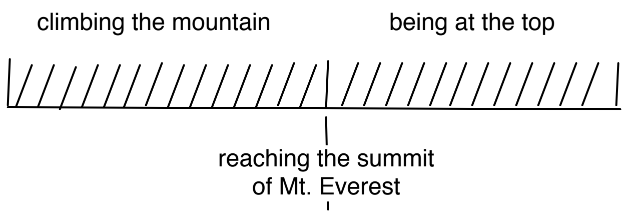

Figure 2.3: Tripartite event ontology by Moens and Steedman (MS88) with Example

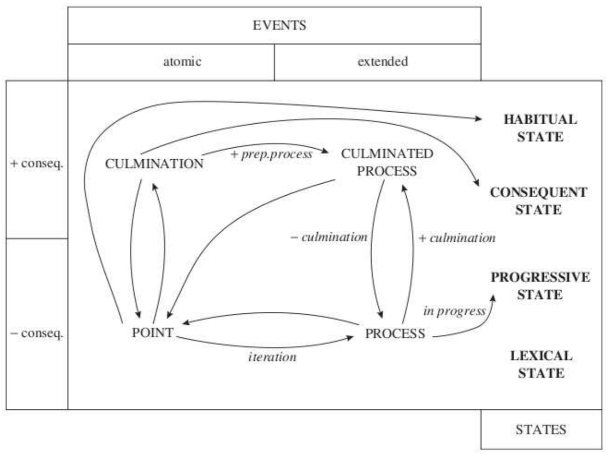

Figure 2.4: Transition network for Moens and Steedman's (MS88) aspectual class

In our work for identifying temporal relations (section 4.5), we use the event properties and classifications (section 3.1.1, 7.1.2) that are influenced by the above mentioned event classifications.

2.3 Temporal Information in Discourse

Many of the approaches to understanding temporal information discussed in the previous sections assume sentences in isolation without the discourse context. In reality, the discourse context is necessary to understand language, in this case, the temporal ordering of all events in discourse. Consider the example (2.28), due to Webber (Web88). With the proposed approaches in the previous sections, we cannot find the relationships among events in discourse. Webber (Web88) proposes a theory of *anaphoric reference* to handle such instances.

(2.28). (a) John went into the florist shop.
(b) He had promised Mary some flowers.
(c) So he picked out three red roses, two white ones, and one pale pink.

There has been other remarkable work on temporal information in discourse, such as, Dowty's (Dow86) temporal discourse interpretation principle (TDIP); Hwang and Schubert's (HS92) work on *tense tree* to improve Reichenbach's proposal to handle embedded clauses, and to handle discourse context that involves shifts in temporal perspectives; Kamp and Reyle's (1993) (KR93) work on using Discourse Representation Theory (DRT) for *tense* and *aspect*.

In addition to the discourse context, pragmatic principles and background world knowledge seem to play a crucial role in understanding temporal information. Therefore, theories concerning only tense and aspect in discourse are insufficient. Consider the following examples:

(2.29). Tag *stood* up. Kostas *greeted* him.

(2.30). Tag *fell*. Kostas *pushed* him.

(2.31). Naima *opened* the door. The room *was* pitch dark.

(2.32). Naima *switched off* the light. The room *was* pitch dark.

The paired examples: (2.29) and (2.30), or (2.31) and (2.32) have the same syntax. Hence using the compositional semantics one would predict that the events stand in similar temporal relations. In (2.29) the order in which the events are described *matches* their temporal order, whereas in (2.30) the descriptive order of events does not match temporal order. On the other hand, the event and state in (2.31) temporally *overlap*, whereas in (2.32) they do not.

The work of Lascarides and Asher (1993) (LA93) attempts to provide a formal account of the pragmatic influences for event ordering in discourse.

The examples above explain that a complete understanding of temporal information requires the understanding of discourse and pragmatics information. However, in this dissertation, we primarily capture the syntactic and semantic information of language to understand the temporal information. Future computational linguistic approaches should consider the above mentioned work to develop better systems.

2.4 Temporal Logic

The next task after understanding temporal information in language is to computationally *represent* time (temporal logic) and *reason* about time (temporal reasoning). If we have an easy to understand expressive representation, the temporal reasoning (TR) system can perform effectively. Hence, temporal logic is a very important component of TR. We will describe three main ways to represent time in logic: first-order logic with temporal arguments, modal temporal logic and reified temporal logic.

2.4.1 The Temporal Argument

The Temporal Argument (Hau87) method represents time with an extra parameter in a first-order predicate calculus. Functions and predicates are extended with an additional temporal parameter to represent time. For example, "Naushad ate with Neema on February 21^{st}," can be represented by *eat(Naushad, Neema, 02-21)*, in which the date is in MM-DD format.

Temporal Argument (TA) is a natural and simple way to represent time in language, but it does not give any special status to time, making it less expressive than other methods.

To perform temporal reasoning in TA, a temporal ordering relation ($\leq$), along with related axioms and temporal constant t_0 representing present time, has to be introduced.

2.4.2 The Modal Temporal Logic

Another way to represent the concept of time is to extend the language of predicate calculus with modal temporal operator. For example in the statement, "Naushad danced with Neema," the semantic representation should capture that there is a time prior to the speech time when the dancing occurred. It can be stated as, there is a time *t'* before the speech time (t) when *Naushad dances with Neema* is true. Prior (1968) (Pri68) describes this basic analysis for the past tense using Minimum Tense Logic K_t. In his analysis, it would be as follows:

$\phi =$ *Naushad dances with Neema*

$P\phi$ is true at time t iff there is a *t'* prior to t such that ϕ is true at time *t'*.

Here, $P\phi$ clearly represents *Naushad danced with Neema* considering speech time as t. Prior represents future tense in similar manner as:
$F\phi$ is true at time t iff there is a t' after t such that ϕ is true at time t'.

Except for a few cases, Prior's (Pri68) Tense Logic can match well with Reichenbach's (Rei47) use of Tense, which has been used by many applications. Table 2.2 (due to (MPG05)) compares Reichenbach's and Prior's proposals.

Relation	Reichenbach	Prior	English Tense	Example
S > R > E	Anterior Past	$PP\phi$	Past Perfect	*I had slept*
S > R = E	Simple Past	$P\phi$	Simple Past	*I slept*
S > R < E	Posterior Past	$PF\phi$		*I would slept*
S = R > E	Anterior Present	$P\phi$	Present Perfect	*I have slept*
S = R = E	Simple Present	ϕ	Simple Present	*I sleep*
S = R < E	Posterior Present	$F\phi$	Simple Future	*I am going to sleep*
S < R > E	Anterior Future	$FP\phi$	Future Perfect	*I will have slept*
S < R = E	Simple Future	$F\phi$	Simple Future	*I will sleep*
S < R < E	Posterior Future	$FF\phi$		*I shall be going to sleep*

Table 2.2: Comparison of Reichenbach's and Prior's proposals

Along with the temporal operators P and F, two additional operators H (for "has always been") and G (for "is always going to be") are also specified. The logical system K_t consists of the following four axioms:

(2.33). $\phi \rightarrow HF\phi$: What is, has always been going to be;

(2.34). $\phi \rightarrow GP\phi$: What is, will always have been;

(2.35). $H(\phi \rightarrow \psi) \rightarrow (H\phi \rightarrow H\psi)$: Whatever always follows from what always has been, always has been;

(2.36). $G(\phi \rightarrow \psi) \rightarrow (G\phi \rightarrow G\psi)$: Whatever always follows from what always will be, always will be.

One short-coming of Minimal Tense Logic (MTL) is not being able to represent the absolute precise time. To handle this short-coming, an extension is proposed to incorporate operator *AT(t)*, where t is a temporal constant, and $AT(t)\phi$ expressing that the proposition ϕ is true at time t.

In terms of expressiveness, MTL seems to be better than TA, at least for some statements. For instance, *Naushad will have danced with Neema*, would be represented in TA by:

$\exists t.\ now < t \wedge \exists t_1.\ t_1 < t \wedge dance(Naushad, Neema, t_1)$

This representation is hard to understand. MTL representation simplifies it by:

$(P(F\ dance(Naushad, Neema)))$

Another benefit of MTL is modularity. It can easily combine other modal qualifications such as *belief, knowledge, etc.* However, one disadvantage of MTL is not being very efficient, as theorem proving in modal logic is more difficult than in first-order logic. Another disadvantage of MTL is that it is not as expressive as Reified Logic.

2.4.3 The Reified Temporal Logic

Reified approaches use first order logic but with higher expressive power that allows one to discuss the truth of assertion. Reifying a logic requires moving into a meta-language, in which a formula in the initial language becomes a *term* in the new language. Hence in Reified logic, one can reason about the particular aspects of the truth of expressions of the object (initial) language through the use of truth predicates. First-order logic is usually taken as the object (initial) language - although modal logic can also be used as the object language as well.

Reified Temporal Logic expresses *when* things are true. The *truth predicate* takes a formula in the initial language (e.g. first-order logic) as one argument, and takes a temporal object as another argument. For “Naushad ate with Neema on February 21^{st},” we can represent the non-temporal part, “Naushad ate with Neema,” in first-order logic as, $eat(Naushad, Neema)$. Hence, in Reified Temporal Logic, we have a *truth predicate*, $HOLDS$, which has one argument as $eats(Naushad, Neema)$, and another argument as the temporal object – $February\ 21^{st}$., i.e. $HOLDS(eats(Naushad, Neema), 02 - 21)$.

In this example, one could mean that Naushad and Neema have been eating the whole day, or at a single time during the day, or at different times during the day. The *truth predicate* is not only used to express when something is true, but also the pattern of its truth, i.e. it is *true* during the whole time or at a sub-interval or any other pattern of the temporal expression. This was not possible in Temporal Argument or Modal Temporal Logic. These properties add another dimension of *time* and make it very expressive, which is why the most influential work used reified temporal logic (All83), (McD82).

2.5 Temporal Primitives

In this section, we explore the temporal reference, or time expression and relation. The first question to ask is what would be primitive for the ontology of time, *time intervals* (periods) or *time points* (instants).

Earlier representations of time considered *instants* or *points* as primitive, as in Situation Calculus (MH82), McDermott (1982) (McD82), and also later Vilain and Kautz et al. (1986, 1990) (VK86), (VKvB90), Miller and Schubert (1990) (MS90), and others. On the other hand, the use of *intervals* better mimics the way humans generally conceptualize time. Considering this, Allen (All83), (All84) proposes his influential work using *intervals* as primitive. We cite some examples from (All83) to understand why *interval* was preferred by Allen.

(2.37). We found the letter *at twelve noon.*

(2.38). We found the letter *yesterday.*

(2.39). We found the letter *while John was away.*

(2.40). We found the letter *after we made the decision.*

In (2.37), *"at twelve noon"* refers to a precise point in time at which the finding event occurred (or was occurring). In (2.38), *"yesterday"* refers to an interval in which the finding event occurred. One thing to note is, both *"twelve noon"* and *"yesterday"* refer to a date system, which are explicit times. In general, the references to temporal relations are both implicit and vague. In particular, the majority of temporal references are implicitly introduced by tense and by the description of how events are related to other events. For example, in (2.39), *"while John was away"* or in (2.40) *"when the decision was made,"* cannot refer to a time point, which is very common in English. Hence, Allen (All83) prefers Temporal *Interval* Algebra instead of *Point* Algebra. Details on Interval Algebra versus Point Algebra can be found in the following sections.

2.6 Temporal Relations

There is a trade-off between expressiveness and efficiency when considering the temporal relations and expressions. We can distinguish the temporal relations in two classes: Qualitative temporal relations and Metric Relations. In this section, we discuss these classes.

2.6.1 Qualitative Relations

Allen (All83) develops an Interval Algebra with thirteen basic (binary) interval relations, in which six are inverses of the other six, excluding equality. These relations are mutually exclusive and they may exist between two intervals. Following are 13 binary interval relations, which are also shown pictorially in Table 2.3:

a. before (<), after (>);

b. overlap (o), overlappedBy (oi);

c. start (s), startedBy (si);

d. finish (f), finishedBy (fi);

e. during (d), contains (di);

f. meet (m), metBy (mi);

g. equality (=)

These temporal relations are transitive, e.g. If A *before* B and B *before* C, using the transitive property of *before*, we can easily conclude that A *before* C. Allen proposes a 13x13 transitive table that models the transitive behavior of all Allen relations. The transitive property of temporal relations is very useful for extracting temporal relations among all possible intervals, as most of the relations are implicit and vague.

Allen proposes this by maintaining a graph of temporal relations among intervals, where the nodes are the intervals and the arcs are labeled by arbitrary disjunctions over the thirteen basic Allen relations. Allen assumes that the network always maintains complete information about how its intervals could be related. When a new temporal relation between two intervals is added, all consequences are generated by computing the transitive closure of the temporal relations. Each new fact adds a constraint about how its two intervals could be

Allen's Temporal Relation	Symbolic Representation	Pictorial Example
X *equal* Y	$X = Y$	XXX YYY
X *before* Y	$X < Y$	XXX YYY
X *after* Y	$X > Y$	YYY XXX
X *meets* Y	X m Y	XXXYYY
X *met by* Y	X mi Y	YYYXXX
X *starts* Y	X s Y	XXX YYYYYY
X *started by* Y	X si Y	YYYYY XXX
X *finishes* Y	X f Y	XXX YYYYY
X *finished by* Y	X fi Y	YYYYY XXX
X *during* Y	X d Y	XXX YYYYY
X *contains* Y	X di Y	XXXXX YYY
X *overlaps* Y	X o Y	XXX YYY
X *overlapped by* Y	X oi Y	YYY XXX

Table 2.3: Allen's Temporal Interval Relations

related, which may introduce new constraints between other intervals through the transitivity rules governing the temporal relations. Allen proposes this constraint propagation algorithm with time complexity $O(N^3)$, where N is the number of intervals. It is linear to add one arc to the network, and the number of modification that can be made is 13 times the number of binary relations between all nodes, which is $O(N^2)$, specifically $13*\frac{1}{2}*(N-1)*(N-2)$.

A problem with Allen's constraint propagation algorithm is that even though it does not generate inconsistencies, it does not detect all inconsistencies in its input. Which means, it is *sound* but not *complete*. The constraint propagation algorithm never compares more than three arcs at a time, and Allen (1983) (All83) shows by example that there could be temporal networks where each subgraph of three arcs is consistent without having a consistent labeling for the whole graph. If complete checking is incorporated, this algorithm becomes exponential, making it *intractable*. The main problem is that it does not restrict the labels on the arcs; any disjunction of basic relations is allowed, i.e. it admits many possible relations between intervals (2^{13} relations to be exact).

Vilain, Kautz and van Beek (1986, 1990) (VK86), (VKvB90) propose a *tractable* solution by restricting to a subset of Interval Algebra using Point Algebra instead of Interval Algebra. They use 181 possible relations (All91) instead of $2^{13} = 8192$ relations of Allen.

Allen (1983) (All83) shows that intervals can be represented with *points* as well. For example, interval X can be represented with *x1, x2*, where *x1* is the start point and *x2* is the end point and *x1* < *x2*[2]. All the basic Allen relations can be rewritten using a pair of points.

Back to the Point Algebra discussion – four *point relations* between the beginning and the end of two intervals are used to define the Point Algebra.

[2]Allen shows it for interval t with the start point *t-* and the end point *t+*.

Any basic relation between intervals can be represented by defining the four relations R1 through R4 as shown in Figure 2.5.

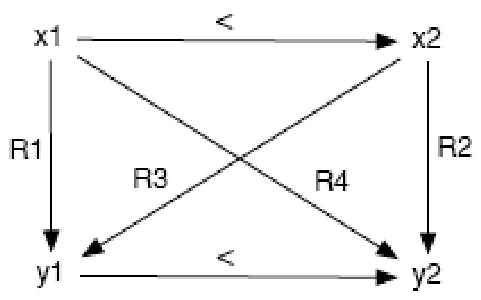

Figure 2.5: Decomposing an interval relation

The labels R1 through R4 on the point relations are taken from the set $\{<, =, >\}$, so there are only three basic relations instead of thirteen. A table for convex relation[3] is given in Figure 2.6. There is a mapping from convex relation to the disjunctions of interval relations, but convex relations cannot express all the disjunctions of interval relations. For example, $x2 \geq y1$ maps to the disjunction $\{> m\}$, but the convex relation does not cover the disjunction $\{< si >\}$.

The point algebra of convex relations can be easily mapped to a subset of Allen's interval algebra using Table 2.6. This smaller subset of interval algebra with 181 possible labels, rather than $2^{13} = 8192$ labels, now makes detecting inconsistencies tractable. Vilain et al. (1990) (VKvB90) prove that Allen's constraint propagation algorithm is sound and complete if the reduced set of labels is adopted. In general, Allen Interval Algebra (All83) is more expressive and flexible, but Point Algebra (VK86) is more efficient.

[3]A convex relation is a relation between four points where $\{<\}, \{=\}, \{>\}, \{\leq\}, \{\geq\}$ labels are only allowed

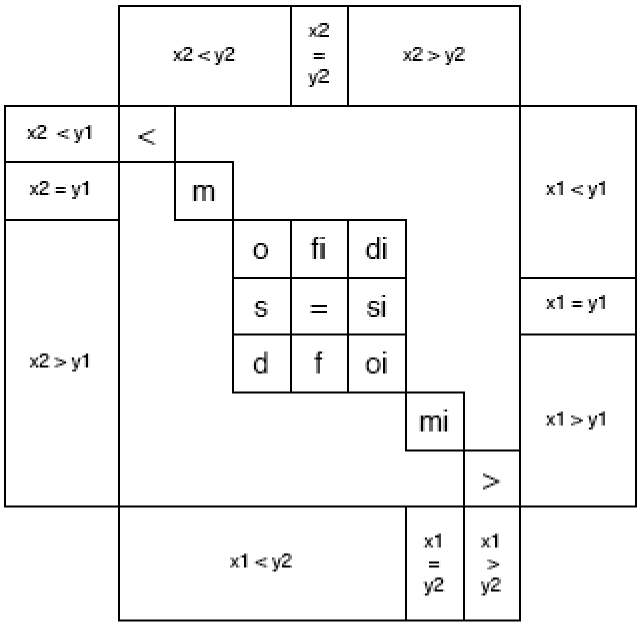

Figure 2.6: Mapping between Interval relations and Point relations

2.6.2 Metric Relations

If time reference is available as a date or in a precise numeric form then we have a numeric absolute temporal reference. We can timestamp the assertion in absolute numeric values. The durations can be numerically represented and easily computed by subtracting numerical values and we have a constant time algorithm that efficiently answers queries about occurrences. However the problem is that we need to have the time reference as a date or in a precise numeric form, which is not available in many scenarios. But there are also many applications, where the presence of time reference is a reasonable assumption, e.g. applications involving real-time data gathering, databases of transactions on a single machine, a central machine maintaining banking records, or a scheduler, and many more.

However, it is not always possible to get the precise time reference for all instances. Usually the temporal distance between events is not available. In such cases, a useful representation proposed by Dean and McDermott 1987 (DM87) as TimeMap is an *acyclic directed graph* (nodes represent events and arcs represent distances between nodes). This representation maintains a partial ordering between events. In TimeMap, events are represented by a (lower bound, upper bound) constraint. These bounds of the unknown events are computed by adding distances between known nodes. If there are several possible paths to an unknown node, then the minimal path distance is the lower bound and the maximum path distance is the upper bound. New available information can be added as temporal constraints that can change distance bounds. An example of a TimeMap is shown in Figure 2.7 (due to (Vil94)).

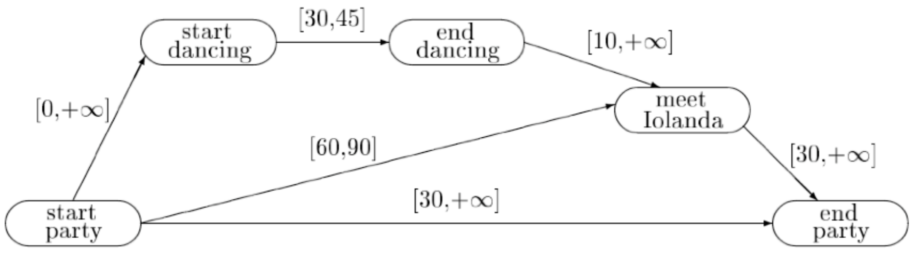

Figure 2.7: An Example of TimeMap

Timegraph: Another representation of metric relations is Timegraph (MS90). Timegraph is explained in detail in section 5.3.1.

2.6.3 Qualitative and Metric Relations

Qualitative and metric relations can be integrated in a single system. Kautz and Ladkin ((Kau91), (KL91)) implement one such system. They keep metric and qualitative interval based components separated and connect them with an inter-component relationship. The reasoning task is done by solving each component separately and then translating and combining the components to converge to a global solution.

2.7 Temporal Reasoning

A significant number of applications can be implemented with the methods described in previous sections. In this section we summarize the existing temporal reasoning systems, their approaches and compare among them.

2.7.1 Temporal Reasoning Systems

In this section we discuss and compare different Temporal Reasoning systems based on the study done by Yampratoom and Allen (YA93). Yampratoom and Allen (YA93) compare among TimeLogic by Koomen(Koo89), MATS by Kautz and Ladkin (Kau91) (KL91), Time Graph by Miller and Schubert (MS90), Time Graph II by Gerivini and Schubert et al. (GSS93), Tachyon by Arthur and Stillman (AS92), and TMM by Dean and McDermott (DM87) (Sch89). These systems vary considerably in underlying mechanism, expressive power, and scale up ability. Following is a brief overview of each system:

TimeLogic: TimeLogic by Koomen (Koo89) is an interval based system. It performs transitive closure computations on intervals. It uses automatically created reference intervals to limit constraint propagations. It cannot handle metric information.

MATS: MATS by Kautz and Ladkin (Kau91), (KL91) handles both quantitative (metric) and qualitative (Allen) constraints, including *disjunction* constraint. Allen constraints are interval based and metric constraints are point based. MATS performs transitive closure computations on intervals for

Allen constraints, and uses constraint satisfaction for point based metric reasoning.

Time Graph: Time Graph by Miller and Schubert (MS90) also handles both metric and qualitative information, but cannot handle *disjunction* relationship. It uses a partial order graph, in which nodes represent time points. Directed links between points represent relations between points ($<$ or $\leq$). The timegraph is partitioned into chains with possible links between chains.

Time Graph-II: Time Graph II by Gerevini and Schubert et al. (GSS93) is the successor of Time Graph. It also uses the *timegraph* data structure. Time Graph II supports a full set of point relations ($<$, $=$, $\leq$, $\neq$). Time Graph II automatically structures the *timegraph* for efficiency. Time Graph II does not handle metric information, which was handled in Time Graph.

Tachyon: Tachyon by Arthur and Stillman 1992 (AS92) can handle both metric and qualitative information, but it simplifies the *disjunction* constraints by presenting the first found solution. This feature does not produce the most general solution as done by other systems. Hence, (YA93) did not use this feature in their evaluation. Tachyon performs constraint satisfaction for point based metric reasoning and translates qualitative constraints into quantitative ones by introducing the concepts of *epsilon* and *infinity*.

TMM: TMM by Dean and McDermott (DM87) and (Sch89) is a temporal database management system. It uses temporal constraint graph (TCG) data structure to construct a time map with single point relations ($<$, $=$, $\leq$). It can handle both metric and qualitative information but cannot handle *disjunction* relationship.

2.7.2 Comparison of different Temporal Reasoning Systems

The differences of these systems are evident in the descriptions. Table 2.4 additionally gives a good overview of the differences. Because of these differences and due to the different degree of optimization done on these systems, an absolute comparison is not useful. Since these systems do not exactly have the same expressivity, Yampratoom and Allen (YA93) consider measurements on *assertion time*, *query time* and how well systems *scale up* to the growing amount of data to compare the performances. In the next section we discuss how these systems perform in practice.

2.7.3 Performance of Temporal Reasoning Systems in Practice

To evaluate different temporal reasoning (TR) systems, Yampratoom and Allen (YA93) generate a database of train schedules with temporal constraints to schedule the trains. Their experiments with different sized datasets show that *for large temporal datasets – where information is added incrementally throughout the execution of the program – systems using incompletely connected graphs (TMM, Time Graph and Time Graph II) seem to be the best options.* Even though these systems do not offer the constant query time, the saving at assertion time is so substantial that the relatively small performance penalty for queries is a reasonable tradeoff. On the other hand, *these systems do not offer the expressiveness since they are using the point based system, instead of the interval based one.* Among these systems, Time Graph II handles a wider range of point based relations, as shown in Table 2.4. But unlike TMM and Timegraph, Time Graph II does not handle the metric relations. Hence, the decision among these three system are determined by their

Underlying Mechanism	
Criteria	Systems using constraint satisfaction problem at assertion time and fixed query time
System	TimeLogic, MATS and Tachyon
Criteria	Systems that build graph structures and may require a search at query time to compute an answer
System	Time Graph, Time Graph II, TMM
Expressive Power	
Criteria	Disjunctive information about time intervals
System	TimeLogic and MATS
Criteria	Supports simple point relations $<$, $=$, $\leq$
System	Time Graph and TMM
Criteria	Supports full point relations $<$, $=$, $\leq$, $\neq$
System	Time Graph II
Quantitative/Quantatitive Information	
Criteria	Handles only qualitative constraint
System	TimeLogic and Time Graph II
Criteria	Handles quantitative constraint and qualitative (without *disjunction* relationship)
System	TMM, Time Graph and Tachyon
Criteria	Handles both qualitative and quantitative constraint
System	MATS
Persistence of Facts	
System	TMM
Implementation Language	
Language	C++
System	Tachyon
Language	Lisp
System	Rest

Table 2.4: Comparison of different Temporal Reasoning system

reasoning capabilities rather than raw performance.

On the other hand, if the assertion time is not an issue – e.g. in an application where the database can be constructed in advance – then the constraint satisfaction approaches (MATS and TimeLogic) can be used. It requires a huge memory for the final database, as these systems are fully connected graphs. Additionally, the performances of MATS and TimeLogic are so unsatisfactory in large datasets that these systems are not practically usable, even if the database is constructed in advance. According to (YA93), these systems might be paying unnecessary penalty due to garbage collection, so they suggest the development of a new algorithm that does not create so many temporal objects.

If we consider a small dataset, then all the systems can be used effectively. The choice can be determined by the expressivity and other reasoning facilities that each TR system provides. For example, unlike other systems, TMM can reason about the persistence of facts. This type of reasoning is very critical for database applications. Hence, TMM would be a good choice for these applications. As another example, the interval-based relations have been useful in planning algorithms and in natural language applications. Therefore, if the dataset is small, MATS and TimeLogic could be effectively used in these applications. On the other hand, if the execution time is still critical, the tradeoff must be considered between the assertion and the query time.

In section 5.4, we describe our system for temporal question-answering with temporal reasoning. We consider the Timegraph (MS90) out of all the reasoning systems since we want a system to handle large documents with faster operations. The additional expressiveness of disjunctions, which is missing in the Timegraph, does not contribute much in our applications.

2.8 Summary

First, we present the linguistic theories on *tense* and *lexical aspect* for understanding temporal information. Next, we present the Artificial Intelligence theories of computational approaches on representing and reasoning temporal information.

We present the influential work of Reichenbach (1947) (Rei47) to understand **tense**. Reichenbach's proposed that *tense* gives information about three types of *times*: (i) the time of speech, (ii) the reference time and (iii) the time of event/state.

Next, we describe linguistic theories to understand whether an **event** has ended or is ongoing, whether it is happening at a point in time or over a period of time. We also describe Vendler's (1967) (Ven67) *statives, activity, accomplishment* and *achievement* classes. Additionally, we briefly explain Moens and Steedman's (1988) (MS88) extension on Vendler's proposal. Finally, we briefly discuss about the influences of discourse and pragmatics on identifying temporal ordering of events in language.

In the next sections, we present Artificial Intelligence theories to represent and reason about temporal information. We start by describing the **Temporal Logic** with possible solutions of Temporal Arguments (Hau87), Modal Temporal Logic (Pri68) and Reified Temporal Logic. Next we explore the **temporal primitives** - *time interval* & *time points* and **temporal relations**. Finally, we conclude by describing the existing approaches for **temporal reasoning**. We compare among temporal reasoning approaches and summarize which approach is more appropriate for a specific scenario.

3 Related Corpus Linguistics Research

In this chapter we describe the corpus linguistics solutions for temporal information understanding, which has been the dominant approach in recent years.

The methodologies described in the previous chapter to understand, represent and reason about *temporal information*, use approaches from analytical linguistics and symbolic AI. In recent years, however, with the rise of computational power and the availability of a wide range of texts, researchers shifted more towards analyzing the data with corpus linguistics solutions to solve natural language problems.

Here, we cite some benefits of the corpus linguistic approach from McEnery and Wilson (1996) (MW96):

- Corpus guided research reveals both the variety and the distribution of the forms of expression in a real sample of language. The distribution is important for algorithm builders to focus on the most frequently occurring cases, to aid in the construction of probabilistic models, and to guide psycholinguistics or cognitive psychological research.
- The annotation schemes, and the *corpora* which is annotated according to the schemes, together provide objective data resources that can

be shared, argued over, and refined by the computational linguistics community. The comparison among the annotations done by human annotators according to the same scheme, can be used to decide how well-defined and comprehensive the scheme is.

- Annotated corpora are resources that can be exploited by machine-learning algorithms to acquire annotation capability without the necessity of implementing an underlying analytical model.
- Annotated corpora provide an objective basis to evaluate competing algorithms.

In the next sections of this chapter, we describe the temporal annotation schemes, the existing systems, and their approaches of using the annotated corpus to extract temporal information from text.

3.1 Annotating Temporal Information

3.1.1 Annotating Events and States

An event is usually a cover term for situations that occur at a specific time or last for a period of time (PCI$^+$03). It may be expressed by tensed (example (3.1)) or untensed verbs (3.2), nominalizations (3.3), adjectives (3.4), predicative clauses (3.5), or prepositional phrases (3.6) (PCI$^+$03).

(3.1). A fresh flow of lava, gas and debris **erupted** there Saturday.

(3.2). Prime Minister Benjamin Netanyahu called the prime minister of the Netherlands **to thank** him for thousands of gas masks his country has already contributed.

(3.3). Israel will ask the United States to delay a military **strike** against Iraq until the Jewish state is fully prepared for a possible Iraqi **attack**.

(3.4). A Philippine volcano, **dormant** for six centuries, began exploding with searing gases, thick ash and deadly debris.

(3.5). "There is no reason why we would not **be prepared**," Mordechai told the Yediot Ahronot daily.

(3.6). All 75 people **on board** the Aeroflot Airbus died.

To annotate *events*, we need to decide on:

- *Which semantic types to annotate? Just events, or events and states? What should be the difference between events and states?*
- *Which textual representation of the event/state should be annotated?*
- *Which attributes should be associated with the annotated events/states?*

These are described in details below:

The semantic types to annotate

Most work on annotating event expressions, except TimeML (PCI$^+$03), does not distinguish between events and states. Merging both *events* and states assumes that the distinction between events and states is too difficult or insignificant for the purpose of annotation.

For the short term, it seems reasonable to start with a merged class, however in the long term the merged class would ignore the fact that there are semantic differences between events and states, which would limit the inferences that can be drawn.

Textual representations to annotate

The next question to ask is which text spans should be annotated, i.e. what would be the textual representation of the event? Considering the events conveyed by the verb clauses, one could decide that the entire clause should be annotated (Filatova and Hovy, 2001 (FH01)); or one could decide to annotate just the verb groups or just the heads of verb groups for simplicity (Pustejovsky et al., 2003 (PCI$^+$03)).

Attributes of Events and States

After annotating the events we need to figure out which attributes should be associated with the annotated events/states. This depends on the application for which we are annotating the corpus. Pustejovsky et al. (2003) (PCI$^+$03) associate *tense, aspect, class, modality* and *polarity* information with events for their news corpus. The event *classes* they propose are: occurrence (*crash, merge*), state (*on board, love*), reporting (*say, report*), intensional action (*attempt, offer*), intensional state (*believe, want*), aspectual (*begin, stop*), and perception (*see, hear*). They classify these classes as there are distinct temporal inferences that can be drawn from each of these event classes.

We list all the TimeML event attributes in Table 3.1. Pustejovsky et al. make minor changes in *tense* information for TempEval-2 (VSCP10). We report the updated attributes in Table 3.1.

3.1.2 Annotating Temporal Expressions

The most obvious temporal information to annotate in text are the **temporal expressions** (as found in temporal adverbials); that is, expressions that refer to **times** (*April 7, 2012*), **durations** (*four years*), or **frequencies** (*daily*). Distinguishing these different types of temporal expressions is important to

event attribute	*possible values*
tense	present, past, future, prespart, pastpart, infinite, none
aspect	progressive, perfect, perfect progressive, none
pos	verb, noun, adjective, preposition, other
class	occurrence, state, report, aspectual, perception, intensional state, intensional action
modality	to, should, would, could, can, might, none
polarity	positive, negative

Table 3.1: TempEval Attributes for Event

locate the events in terms of absolute times (calendrical time) or in terms of relation to other events.

One must deal with the following kinds of temporal expressions for annotations:

- Specific calendrical dates, e.g. *26 March 1971, or 21st February 1952, or December 16, 1971*, etc.
- Contextually dependent temporal expressions, such as *now, tomorrow, Sunday*, which have different values at different times, and are interpreted in terms of the speech time or the document creation time.
- Relational expressions relative to time - *first week of summer*, and events - *two days before the New Year.*
- Vague expressions like *several months ago, sometime in the summer*, etc.

There has been significant efforts to annotate and recognize temporal expressions. Recognizing temporal expressions is an integral part of many Information Extraction (IE) shared tasks (e.g. MUC-6 and 7 Named Entity

Recognition tasks, ACE-2004 Event Recognition task). There are a number of annotation schemes, such as Ferro et al. (2003) (FGMW03), Setzer and Gaizaukas (2000) (SG00), and TERN 2004 (Fer04). Among these, the most widely used annotation is TERN 2004 (Fer04). TimeML's (PCI$^+$03) temporal expression annotation is inherited from TERN annotations. Recently it is more widely used because it incorporates other annotations like events and temporal relations.

In TimeML annotation, the text span for temporal expression can be a multi-word entity (unlike events) or a single-word entity (like events). The following grammatical categories are considered as TIMEX [1]:

- Noun (including proper nouns): e.g. *January, Sunday*
- Noun Phrase (NP): e.g. *the morning, Friday night, the last two years*
- Adjective: e.g. *current*
- Adverb: e.g. *recently*
- Adjective or Adverb Phrase: e.g. *half an hour long, two weeks ago, nearly a month ago*

Attributes of Temporal Expression

Each temporal expression in TimeML has an attribute *type*, which could be DATE, TIME, DURATION, or SET. The format of the *value* attribute (another non-optional attribute) is determined by the *type* of TIMEX. For example, the *value* for a DURATION in TimeML should start with 'P', since duration represents *period of time.* Table 3.2 shows some examples of *type* and *value* attributes according to TimeML.

[1]TIMEX is the short-name for temporal expressions in TimeML

Temporal expression	Type	Value
DCT (given):		
March 1, 1998	TIME	1998-03-01
Sunday	DATE	1998-03-01
last week	DATE	1998-W08
mid afternoon	TIME	1998-03-01TAF
nearly two years	DURATION	P2Y
each month	SET	P1M

Table 3.2: Examples of normalized *values* and *types* for temporal expressions according to TimeML

There is another significant attribute *mod*, which is optional and does not exist in most of the temporal expressions. Some examples of expressions with *mod* attributes are listed in Table 3.3.

3.1.3 Annotating Temporal Relations

The next task is to annotate the *relations* between events[2] and temporal expressions or between events and events, given the annotations for the temporal expressions and the events.

Annotating relations between times and events

The time-event relationship can be conveyed **explicitly** or **implicitly**. "Explicitly" means that the relation is explicitly conveyed in a sentence by a prepositional phrase (example (3.7)).

(3.7). Neema *flew* to New York on Friday.

[2] *events* are used loosely to represent both *event* and *state*

	Value	Sample Expressions
Points	BEFORE	*more than a decade ago*
	AFTER	*less than a year ago*
	ON-OR-BEFORE	*no less than a year ago*
	ON-OR-AFTER	*no more than a year ago*
Durations	LESS-THAN	*less than 2 hours long*
	MORE-THAN	*more than 5 minutes*
	EQUAL-OR-LESS	*no more than 10 days*
	EQUAL-OR-MORE	*at least 10 days*
Points and Durations	START	*the early 1960s, the dawn of 2003*
	MID	*the middle of the month, mid-summer*
	END	*the end of the year*
	APPROX	*about three years ago*

Table 3.3: Values of *mod* attribute

(3.8). Naima *arrived* home at 9 p.m. Later she *cooked* noodles and then *ate* all of it.

However, temporal expressions can also be conveyed in implicit ways, (example (3.8)). There is no explicit temporal relation between *cooked* and *9 p.m.*, or between *ate* and *9 p.m.*; but there are explicit event-event relations between *cooked* and *ate*, and between *cooked* and *arrived*, and an explicit event-time relation between *arrived* and *9 p.m.* Therefore, we can infer the implicit temporal relation between *ate* and *arrived* through these explicit relations.

TimeML (PCI+03) captures temporal relations with TLINK (temporal link). TLINK has an *event id* (to identify the event), a *timex id* (to identify the temporal expression) and a *temporal relation.* To annotate "John *taught* on *Monday*", TimeML annotates *taught* as an event (e.g. with event id e1), *Monday* as a temporal expression (e.g. with timex id t1), and <`TLINK eventInstanceID=e1 relatedToTime=t1 relType=IS-INCLUDED/`> as a TLINK.

Initially TimeML selected Allen's temporal relations (described in 2.6.1) and annotated the TimeBank corpus (PHS+03). However, in later annotations for the temporal evaluation shared tasks (TempEval 2007 (VGS+07) and TempEval 2010 (VSCP10)), the creators of TimeML introduced a smaller subset with *before, after, before-or-overlap, overlap-or-after, overlap* and *vague* relations. Their motivation for selecting a smaller set of relations was to make the annotation task easier. However, their inter-annotator agreement was reported 65% (*event-time* relations) and 72% (*event-event* relations) (VGS+07), whereas the inter-annotator agreement for the full set of relations in TimeBank was 77%[3]. The smaller set of relation is also unable to make fine distinctions required for temporal reasoning. TempEval organizers have reverted back to

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

the original relation set for TempEval 2013[4].

Allen interval relations and their equivalent TimeML relations are reported in Table 3.4.

TimeML also annotates the temporal relations between the events and the document creation time.

Annotating relations between events

Like *event-time* relations, *event-event* relations may be conveyed explicitly or implicitly. For *event-event* relations, TimeML uses the same temporal relations as *event-time* relations.

Both TempEval 2007 (VGS⁺07) and TempEval 2010 (VSCP10) annotate temporal relations between the main events of consecutive sentences. TempEval 2010 also introduces intra-sentential event-event relations when one event syntactically dominates the other. For example, there will be a relation between `e2` and `e1` in the sentence, "The students *heard*$_{e1}$ a *fire alarm*$_{e2}$".

Subordinating and Aspectual relations

It is not always possible to annotate temporal relations between all events in a sentence. Take for example, *John might have kissed the girl he met at the party* or *John hoped to kiss the girl he met at the party (and did/did not)*. In both cases, we cannot mark the temporal relation between *kiss* and *meet*, because we do not have the information if it actually occurred or not. TimeML tries to capture these subordinate relations with a SLINK or Subordinate Link. TimeML, however, does not capture all possible subordinate links. It only considers:

(3.9). Modal: Arshad **promised** Jhuma to <u>buy</u> some chocolates.

[4]http://www.cs.york.ac.uk/semeval-2013/task1/

Allen's Temporal Relation	TimeML Relation	Pictorial Example
X *equal* Y	X *simultaneous* Y X *identity* Y X *during* Y	XXX YYY
X *before* Y	X *before* Y	XXX YYY
X *after* Y	X *after* Y	YYY XXX
X *meets* Y	X *ibefore* Y	XXXYYY
X *met by* Y	X *iafter* Y	YYYXXX
X *starts* Y	X *begins* Y	XXX YYYYYY
X *started by* Y	X *begun by* Y	YYYYY XXX
X *finishes* Y	X *ends* Y	XXX YYYYY
X *finished by* Y	X *ends by* Y	YYYYY XXX
X *during* Y	X *is included* Y	XXX YYYYY
X *contains* Y	X *includes* Y	XXXXX YYY
X *overlaps* Y	not represented	XXX YYY
X *overlapped by* Y	not represented	YYY XXX

Table 3.4: Allen interval relations and their equivalent TimeML relations

(3.10). Factive: Neema **forgot** that she was in Rochester last year.

(3.11). Counter-factive: Marzina was **unable** to sleep.

(3.12). Evidential: Akhtar **said** he bought some chips.

(3.13). Negative evidential: Naima **denied** she bought ice cream.

TimeML also annotates a special kind of link, ALINK (aspectual link), to capture the relationship between aspectual event and its argument event. *Aspectual Link* examples include:

(3.14). Initiation: John **started** to read.

(3.15). Culmination: John **finished** assembling the table.

(3.16). Termination: John **stopped** talking.

(3.17). Continuation: John **kept** talking.

(3.18). Reinitiates: John **restarted** the problem from scratch.

3.2 The Existing Corpus Based Systems

With the availability of the annotated corpus with temporal information (PHS$^+$03) and the shared tasks on temporal information processing (VGS$^+$07) (VSCP10), many researchers implemented systems to extract *events*, *temporal expressions* or *identified temporal relations*. In this section, we describe and summarize the current approaches for these automated systems.

3.2.1 Event Extraction

Even before TimeML (PCI$^+$03), there had been significant research done on the *event* extraction task, such as MUC-7[5] or ACE[6] specifications, which produced Ahn [1], Aone and Ramos-Santacruz [4] and many other systems. In those specifications, the task is not just to extract the event, but also to extract the event arguments, assign roles and determine event co-references. However, in these specifications, the entities are limited and use a pre-defined set such as person, organization, location, geo-political entity, facility, vehicle, weapon, etc. In this dissertation, we are interested in all events, as outlined by TimeML, instead of a limited set of events with detailed information. Systems extracting TimeML *events* are described below.

Saurí et al. (2005) (SKVP05) implement an event and event feature extraction system EVITA[7] for automatically extracting TimeML *events*. Their system is a linguistically motivated rule-based system, with some statistical guidance for disambiguation. They consider *morphosyntactic information* (Part-of-speech tags – POS, lemmatizing, lexicon lookup, shallow parsing, etc) and *lexical semantic information* (WordNet synsets) to build their automated system.

Boguraev and Ando (2005) (BA05) present a data-driven approach, which is trained on TimeBank corpus (PHS$^+$03). Their system is based on *morphosyntactic features* and uses a Robust Risk Minimization (RRM) classifier.

Bethard and Martin (2006)'s (BM06) apply machine learning classifier SVM (Support Vector Machine) on the Timebank corpus to build models

[5]Message Understanding Conference

[6]http://www.nist.gov/speech/tests/ace/

[7]EVITA is a module of TARSQI Toolkit (Temporal Awareness and Reasoning Systems for Question Interpretation). Details on TARSQI Toolkit and EVITA system can be found in <`http://www.timeml.org/site/tarsqi/index.html`>

for event extraction and event class classification. For this classification task, they consider *morphosyntactic features* (text features, morphological features, POS, syntactic-chunk label, time chunk label, modifying temporal, etc) and *lexical semantic features* (WordNet hypernym feature).

Edinburgh-LTG (Grover, et al., 2010) (GTAB10) is a rule-based system for TempEval-2 (VSCP10) which uses *morphosyntactic knowledge* and *lexical semantic information*. Edinburgh-LTG uses WordNet hyponyms of event and state concepts, and a lexicon of event triggers which is derived from the training data.

JU-CSE-TEMP (Kolya, et al., 2010) (KEB10) is a rule-based system with *morphosyntactic information* to extract events. This system is also developed for TempEval-2.

TIPSem and TIPSem-B (Llorens et al., 2010) (LSNC10) implement CRF classifiers to extract events in TempEval-2. Both of these systems use *morphosyntactic* features. TIPSem additionally uses *lexical semantic* features and *sentence-level semantic* features. Both TIPSem and TIPSem-B have the current state-of-the-art performance on event extraction.

Summary of *event* extraction on TimeML annotated corpora: We have reported approaches for automated event extraction on TimeBank (PHS$^+$03) and TempEval-2 (VSCP10) corpora. Both of these corpora have insignificant differences for *event* annotation, which make the performances reported in both of these corpora comparable. In Table 3.5, we compare all the systems in terms of their strategies (*rule-based approach* vs *data-driven approach*) and the linguistic information that they used. The detailed performances can be found in Verhagen et al. (2010) (VSCP10). We find by comparing the performances that systems with both *rule-based* (Edinburgh-LTG) and *data-driven* (TIPSem and TIPSem-B) approaches achieve top performances. The performance mainly depends on the linguistic information used by the system. The best performing system, TIPSem, uses *morphosyntactic, lexical semantic and sentence-level semantic* information, and competitive systems use *morphosyntactic and lexical semantic* information. However, one exception is TIPSem-B, which achieves top performance with only *morphosyntactic* information.

Strategy	System	Linguistic Knowledge
Rule-based	EVITA	*ms*[a]
	Edinburgh-LTG (GTAB10)	*ms, ls*[b]
	JU-CSE-TEMP (KEB10)	*ms*
	TRIPS (UA10)	*ms, ls, ss*[c]
Data-driven	Boguraev and Ando (2005) (BA05)	*ms*
	Bethard and Martin (2006) (BM06)	*ms, ls*
	TIPSem-B (LSNC10)	*ms*
	TIPSem (LSNC10)	*ms, ls, ss*
Hybrid	TRIOS (UA10)	*ms, ls, ss*

[a]*morphosyntactic information*, e.g. POS, lexical information, morphological information
[b]*lexical semantic information*, e.g. WordNet synsets
[c]*sentence-level semantic information*, e.g. Semantic Role labels

Table 3.5: Automated approaches for Event Extraction

3.2.2 Temporal Expression Extraction

Earlier systems that extracted temporal expressions were developed in the scope of ACE Temporal Expression Recognition and Normalization (TERN) task (Fer04), in which TIMEX2 tag denotes temporal expressions. There are a few differences between TimeML's (PCI$^+$03) TIMEX3 and TERN's TIMEX2;

notably, TIMEX2 includes post-modifiers[8] (prepositional phrases and dependent clauses) that introduce a related event, but TIMEX3 does not. But to a larger extent TIMEX3 is based on TIMEX2. TempEval-2 (VSCP10) had a shared task on temporal expression extraction, which resulted many systems to extract TIMEX3.

Ahn et al. (2005) (AAdR05), Hachioglu et al. (2005) (HCD05) and Poveda et al. (2007) (PST07) use a token-by-token classification for temporal expressions, which is represented by a B-I-O encoding with lexical and syntactic features. They evaluated on the TERN dataset.

Boguraev and Ando (2005) (BA05), and Kolomiyets and Moens (2009) (KM09) report performances on the recognition of temporal expressions using TimeBank as an annotated corpus. Boguraev and Ando's work is based on a cascaded finite-state grammar (500 stages and 15000 transitions). On the other hand, Kolomiyets and Moens first filter certain phrase types and grammatical categories as candidates for temporal expressions, and then apply Maximum Entropy classifiers.

HeidelTime (Strötgen and Gertz, 2010) (SG10), TERSEO+T2T3 transducer (Saquete, 2010) (Saq10), USFD2 (Derczynski and Gaizauskas, 2010) (DG10), Edinburgh-LTG (Grover, et al., 2010) (GTAB10), and JU-CSE-TEMP (Kolya et al., 2010) (KEB10) participated in TempEval-2 as rule-based systems to extract and normalize *temporal expressions*.

Other participants of TempEval-2, TIPSem, TIPSem-B (Llorens et al., 2010) (LSNC10) and KUL (Kolomiyets and Moens, 2010) (KM10) apply machine learning algorithms to extract *temporal expressions* and use rule-based systems to normalize them. Kolomiyets and Moens (2010) use a maximum entropy classifier, and Llorens et al. (2010) use a CRF (Conditional Random

[8]For the phrase, *five days after he came*, TIMEX2 will annotate the whole phrase as the temporal expression, whereas, TIMEX3 will only capture *five days*.

Field) classifier.

Summary of *temporal expression* extraction on TimeML annotated corpora: We have reported approaches for automated temporal expression extraction on TempEval-2 (VSCP10) and TERN (Fer04) corpora. Due to the completeness and the improvements introduced by TimeML, it is currently considered as the standard annotation scheme. Hence, we summarize the automated systems on TimeML annotation only.

In Table 3.6, we compare all the systems in terms of their strategies (*rule-based approach* vs *data-driven approach*) to extract *temporal expression*. All of these systems use rule-based approaches to normalize *temporal expressions*. The detailed performances can be found in Verhagen et al. (2010) (VSCP10). We find that systems with both *rule-based* (HeidelTime) and *data-driven* (TRIOS) approaches can achieve top performances. HeidelTime (Strötgen and Gertz, 2010) (SG10) currently has the state-of-the-art performance on *temporal expression* extraction and normalization task.

Strategy	System
Rule-based	Boguraev and Ando (2005)
	HeidelTime (Strötgen and Gertz, 2010) (SG10)
	TERSEO+T2T3 transducer (Saquete, 2010) (Saq10)
	Edinburgh-LTG (Grover, et al., 2010) (GTAB10)
	USFD2 (Derczynski and Gaizauskas, 2010) (DG10)
	JU-CSE-TEMP (Kolya et al., 2010) (KEB10)
Data-driven	Kolomiyets and Moens (2009) (KM09)
	TIPSem, TIPSem-B (Llorens et al., 2010) (LSNC10)
	KUL (Kolomiyets and Moens, 2010) (KM10)
	TRIOS, TRIPS (UzZaman and Allen, 2010) (UA10)

Table 3.6: Automated approaches for Temporal Expression Extraction

3.2.3 Temporal Relation Identification

Temporal relations identification task is mostly driven by the availability of the annotated corpus for this particular task. Previously, most work focused on identifying the temporal relations between events (MVW$^+$06), (CWJ07). Later TempEval 2007 (VGS$^+$07) introduced three tasks: intra-sentential event-timex, event-dct[9], and inter-sentential event-event relations. Finally, TempEval 2010 (VSCP10) introduced intra-sentential event-event relations when one event syntactically dominates the other.

In this section, we describe how researchers approached these temporal ordering identification tasks.

Earlier work on TimeML annotated corpus: Earlier work on temporal relation identification used TimeBank corpus, where the systems classified full-sets of temporal relations, which include *before, immediately–before, includes, begins, ends*, *simultaneous* and their inverses.

Mani et al. (2006) (MVW$^+$06) use a maximum entropy classifier on TimeBank corpus to classify temporal relations between pair of events. They take the attributes of events from the TimeBank corpus.

Chambers et al. (2007) (CWJ07) train a Naive Bayes classifier that classifies relations between pairs of events. Initially, they train classifiers to automatically learn TimeBank *event attributes (class, tense, aspect, modality, polarity).* They use the learned *event attributes*, *morphosyntactic features* (POS, lemma, POS bigram, syntactic information, etc.), and *lexical semantic features* (WordNet synsets) to classify temporal relations between a pair of events.

Work on TempEval 2007 (VGS$^+$07): The temporal relation between two entities depends on the types of entities being considered. Considering this, TempEval 2007 (VGS$^+$07) categorize the temporal relation identification

[9]dct is document creation time

task into three subtasks. These subtasks are:

(A) *Temporal relations between an event and a time expression in the same sentence.*

(B) *Temporal relations between an event and the document creation time.*

(C) *Temporal relations between the main events of adjacent sentences.*

For simplification, TempEval 2007 does not consider the full set of temporal relations, instead it uses *before, after, overlap, before-or-overlap, overlap-or-after* and *vague.*

WVALI (Puscasu, 2007) (Pus07) and XRCE-T (Hagege and Tannier, 2007) (HT07) are rule-based systems which use morphosyntactic information. They include timex and event attributes as semantic information.

CU-TMP (Bethard and Martin, 2007) (BM07), NAIST (Cheng, Asahara and Matsumoto, 2007) (CAM07), and USFD (Hepple, Setzer, and Gaizauskas, 2007) (HSG07) are data-driven systems, which also use *morphosyntactic information* and event and timex attributes. CU-TMP uses a support vector machine (SVM) classifier, USFD uses two ML techniques (SVM and Naive-Bayes), and NAIST uses an ML technique that combines SVM and hidden Markov models.

Finally, LCC-TE (Min, Srikanth and Fowler, 2007) (MSF07) is a hybrid system which combines rules and different ML techniques (SVM, Naive-Bayes, maximum entropy, and decision trees). It also uses lexical semantics (*lexical semantic information* and word sense disambiguation techniques.

Out of these systems, WVALI performs the best in the TempEval 2007. Yoshikawa et al.'s (YRAM09) system outperforms all the TempEval 2007 systems in a later published work. Their implementation is based on Markov Logic Network (MLN), but their main focus is to approach all three problems together, instead of handling them in isolation. Their approach of jointly doing all three tasks is discussed in the next section.

Work on TempEval 2010 (VSCP10): TempEval 2010 adds an additional subtask of intra-sentential event-event relations when one event syntactically dominates the other. TempEval 2010 also uses the same relation set as that of TempEval 2007.

USFD2 (Derczynski and Gaizauskas, 2010) (DG10) is a rule-based system that uses *morphosyntactic* information. Derczynski and Gaizauskas highlight the importance of the temporal signals for categorizing temporal relations.

TIPSem, TIPSem-B (Llorens et al., 2010) (LSNC10), NCSU (Ha et al., 2010) (HBLL10), and JU-CSE-TEMP (Kolya et al., 2010) (KEB10) are data-driven systems. TIPSem, TIPSem-B and JU-CSE-TEMP use CRF classifier, and NCSU uses a Markov Logic Network classifier. In terms of features, JU-CSE-TEMP and TIPSem-B consider *morphosyntactic features* and *event and timex attributes*, whereas, NCSU uses event *morphosyntactic features*, *lexical semantic features* and *event and timex attributes*. TIPSem additionally considers *sentence-level semantic features*.

Summary of *temporal relation* identification on TimeML annotated corpora: We have reported approaches for automated temporal relation identification on TimeBank corpus (PHS$^+$03), TempEval-1 (VGS$^+$07), and TempEval-2 (VSCP10) corpus. In Table 3.7, we compare all the systems in terms of their strategies and the linguistic knowledge that they used for *temporal relation* identification task. Although the rule based and the data-driven approaches are both shown to be successful, most systems use the data-driven approach. All of these systems use TimeML *event and timex attributes*, which by itself is not enough. Therefore, most of the systems at least use *morphosyntactic information*. Many systems also use *lexical semantic information*, and a few of them also uses *sentence-level semantic information*.

Corpus and Task	Strategy	System	Knowledge
TimeBank[a]	*data-driven*	Mani et al. (MVW$^+$06) *[MaxEnt]*	*e-attr*[b]
event-event		Chambers et al. (CWJ07)*[Naive Bayes]*	*e-attr, ms*[c], *ls*[d]
TempEval[e]	*data-driven*	CU-TMP (BM07) *[SVM]*	*e-attr, ms*
event-event		NAIST (CAM07) *[SVM and HMM]*	*e-attr, ms*
event-timex		USFD (HSG07) *[SVM and Naive-Bayes]*	*e-attr*
		Yoshikawa et al. (YRAM09) *[MLN]*	*e-attr, ms, g-inf*[f]
		JU-CSE-TEMP (KEB10) *[CRF]*	*e-attr, ms*
		NCSU (HBLL10) *[MLN]*	*e-attr, ms, ls, g-inf*
		TIPSem-B (LSNC10) *[CRF]*	*e-attr, ms*
		TIPSem (LSNC10) *[CRF]*	*e-attr, ms, ls, ss*[g]
		TRIOS, TRIPS (UA10) *[MLN]*	*e-attr, ms, ls, ss*
	rule-based	WVALI (Pus07)	*e-attr, ms*
		XRCE-T (HT07)	*e-attr, ms*
		USFD2 (DG10)	*e-attr, ms*
	hybrid	LCC-TE (MSF07) *[SVM, MaxEnt, Naive-Bayes, Decision Tree]*	*e-attr, ms, ls*

[a]classification on full set of TimeML relations, *before, ibefore, includes, begins, ends, simultaneous* and their inverses.
[b]e-attr: *entity attributes*, e.g. event *class, tense, aspect, polarity, modality*; timex *type, value.*
[c]ms: *morphosyntactic information*, e.g. POS, lexical information, morphological information, syntactic information
[d]ls: *lexical semantic information*, e.g. WordNet synsets
[e]classification on restricted set of relations, *before, after, overlap, before-or-overlap, overlap-or-after and vague*
[f]g-inf: *global inference*, described in the next section
[g]ss: *sentence-level semantic information*, e.g. Semantic Role labels

Table 3.7: Automated approaches for Temporal Relation Identification

3.2.4 Considering Global Inference

All the data-driven approaches, mentioned in the previous section, handle the task of temporal ordering by just considering the entities (events or temporal expression) and their features, and then applying machine learning algorithms. It is obvious that these approaches are completely ignoring the global influence of a specific pair of events, e.g. if we know A *before* B and B *before* C, then for classifying between A and C, we know it should be A *before* C. If we handle this task separately and if a similar probability exists between *before* and *overlap*, then with the global information, we can easily infer that the relation should be *before*. Allen (All83) proposes the 13x13 transitive table that models the transitive properties of all Allen relations (check Allen relations in Table 2.3).

Bramsen et al. (2006) (BDLB06) and Chambers and Jurafsky (2008) (CJ08) use transitive properties to consider the global inference. Bramsen et al. (2006) induce the temporal directed acyclic graph (TDAG). At first they generate the probabilities for pairwise relations produced by local classifiers. To construct TDAG using global inference, they experiment with different strategies like Greedy Inference in Natural Reading Order, Greedy Best-first Inference and Exact Inference with Integer Linear Programming (ILP). They find ILP to perform the best. Following Bramsem et al. (2006), Chambers and Jurafsky (2008) (CJ08) also use Integer Linear Programming (ILP) to handle the global inference, and report 3% improvement on TimeBank corpus for temporal ordering between events. Bramsen et al. use three temporal relations: *forward, backward, and null or not connected*, which are equivalent to *before, after, and vague*. On the other hand, Chambers and Jurafsky use only two temporal relations, *before* and *after*, instead of using either TimeBank(PHS$^+$03)'s 13 temporal relations (derived from Allen (1983) (All83)), or TempEval's coarse grained relations: *before, after, overlap, before-or-overlap, overlap-or-after and vague*.

On the other hand, The system developed by Yoshikawa et al. (2009) (YRAM09) applies ILP in TempEval 2007 data-set and it outperforms all the TempEval 2007 participants. Its implementation is based on Markov Logic Network, but the main focus is to approach all three problems together instead of handling them in isolation. If all the tasks are approached simultaneously then the inference of one task can be benefited by the inference of another task. They use ILP as a base solver in their Markov Logic Network (MLN) framework.

3.3 Evaluating Temporal Information

The evaluation of temporal information can be divided into *entity* evaluation (to evaluate events and temporal expressions) and *relation* evaluation. In this section, we describe the existing evaluation metrics used to evaluate the temporal annotations.

3.3.1 Evaluating Entity Extraction

To evaluate temporal entities (*events* and *temporal expressions*), we need to evaluate:

- How many entities are correctly identified.
- If the extents for the entities are correctly identified.
- How many entity attributes are correctly identified.

Traditionally, Information Retrieval (IR) evaluation metric – *precision, recall* and *F score* – are used, to consider the above mentioned properties. The evaluation is straightforward for a single-word entity *event*. However, for multi-word entity *temporal expressions (timex)*, the evaluation is carried out in two different ways: (i) entity-based evaluation, and (ii) token-based evaluation.

Equations 3.1 and 3.2 explain the *entity-based evaluation*.

$$Precision = \frac{|Sys_{entity} \cap Ref_{entity}|}{|Sys_{entity}|} \tag{3.1}$$

$$Recall = \frac{|Sys_{entity} \cap Ref_{entity}|}{|Ref_{entity}|} \tag{3.2}$$

where, Sys_{entity} contains the entities extracted by the system that we want to evaluate, and Ref_{entity} contains the entities from the reference annotation that are being compared.

Two methodologies are explored in the entity based evaluation: relaxed (partial) and strict (exact) matching. In strict matching two strings match completely, e.g. matching "last five days" against "last five days". On the other hand, in relaxed matching there is a partial overlap between two strings, e.g. matching "last five days" against "five days".

Next we explain the *token-based evaluation* with equations 3.3 and 3.4.

$$Precision = \frac{tp}{tp + fp} \tag{3.3}$$

$$Recall = \frac{tp}{tp + fn} \tag{3.4}$$

where, tp is the number of tokens that are part of an extent in both reference and system; fp is the number of tokens that are part of an extent in the system but not in the reference; and fn is the number of tokens that are part of an extent in the reference but not in the system.

The following example illustrates the differences between two evaluation techniques:

Consider a document containing only three temporal expressions, “the beginning of October 1999”, “2005” and “yesterday”. Imagine that two systems automatically extract temporal expressions. Their correct extractions are shown as 1 and their missing extractions are as 0 in Table 3.8.

timex	*annotation1*	*annotation2*
the	0	0
beginning	1	0
of	1	0
October	1	1
1999	1	1
2005	1	1
yesterday	0	1

Table 3.8: Example extraction for two systems

Performances for these two annotations are shown in Table 3.9. The entity-based score provides a better idea of the number of *timexes* recognized, while the token-based score focuses on the number of tokens and does not consider if a system is missing tokens or entire entities.

annotations	*token-based*	*entity-based*
annotation1 score	5/7 tokens	2/3 entities (relaxed match)
annotation2 score	4/7 tokens	3/3 entities (relaxed match)

Table 3.9: Comparing Entity-based vs Token-based evaluation

Both entity-based and token-based evaluations have been used in the literature. TempEval 2010 (VSCP10) evaluates participants using the *token-*

based evaluation, whereas, Boguraev and Ando (2005) (BA05), Kolomiyets and Moens (2009) (KM09), and UzZaman and Allen (2011) (UA11) evaluate their systems using the *entity-based evaluation*. TempEval organizers have updated their evaluation metric for TempEval 3 to consider the *entity-based evaluation*.

In TempEval-2 (VSCP10), the **attribute** performance is reported as attribute accuracy – calculated as the matching attributes out of the matching reference and system entities (equation 3.5).

$$Attribute\ Accuracy = \frac{|\{\forall x \mid x \in (Sys_{entity} \cap Ref_{entity}) \wedge Sys_{attribute}(x) == Ref_{attribute}(x)\}|}{|Sys_{entity} \cap Ref_{entity}|} \tag{3.5}$$

If one annotation has only one matching entity and gets the attribute for that entity correct, then it gets 100% accuracy; similarly, if another annotation has all the matching entities and gets the attributes for all of them correct, then it also gets 100% accuracy. This makes the attribute performances incomparable.

In order to make the comparison easier, another approach (BM06), (UA11) considers the *attribute recall* – calculated as the number of matching attributes and entities out of the total reference entities (equation 3.6), and *attribute precision* – calculated as the number of matching attributes and entities out of the total system entities (equation 3.7). Attribute recall is equivalent to the multiplication of entity recall and attribute accuracy (multiply equation 3.2 and equation 3.5 to get equation 3.6). Similarly, attribute precision is equivalent to the multiplication of entity precision and attribute accuracy.

$$Attribute\ Recall = \frac{|\{\forall x \mid x \in (Sys_{entity} \cap Ref_{entity}) \wedge Sys_{attribute}(x) == Ref_{attribute}(x)\}|}{|Ref_{entity}|} \tag{3.6}$$

$$\textit{Attribute Precision} = \frac{|\{\forall x \mid x \in (Sys_{entity} \cap Ref_{entity}) \wedge Sys_{attribute}(x) == Ref_{attribute}(x)\}|}{|Sys_{entity}|} \tag{3.7}$$

Attribute F-score is generally used (BM06), (UA11) to evaluate entity attributes. However, *Attribute recall* can also be used alone to evaluate the attribute performance (LUA12). This makes a better evaluation, since it does not penalize for attribute precision[10].

3.3.2 Evaluating Temporal Relations

Temporal relation evaluation has two components: *identification* of a pair of entities that has a temporal relation, and *classification* of the temporal relation for a pair of entities. Previous TempEval shared tasks (VGS$^+$07), (VSCP10) ignored the task of identification and assumed the pair of entities. The participants' task was to classify the temporal relations.

The current relation evaluation metric compares the system relations against the reference relations to evaluate the temporal relation classification. The standard precision and recall definitions are used to evaluate the systems (VGS$^+$07), (VSCP10).

$$\textit{Relation Precision} = \frac{|Sys_{relation} \cap Ref_{relation}|}{|Sys_{relation}|} \tag{3.8}$$

$$\textit{Relation Recall} = \frac{|Sys_{relation} \cap Ref_{relation}|}{|Ref_{relation}|} \tag{3.9}$$

[10]If some system extracts many wrong entities, then in attribute precision, the system gets penalized for that. These systems are already getting penalized in the entity precision, hence to evaluate how a system can extract an entity attribute, we want to judge the attributes' performance only, instead of penalizing for entity precision.

where, $Sys_{relation}$ contains the relations identified by the system that we want to evaluate, and $Ref_{relation}$ contains the relations from the reference annotation that we are comparing against.

Since the pair of entities are assumed, the only task is to classify the temporal relation between the entities of that pair. As a result, the *precision* and *recall* are the same in both cases and it is more appropriate to call it *accuracy*. However, a few participants (XRCE-T (HT07) and TRIOS/TRIPS (UA10)) extracts the entities and their attributes from raw text, and then classifies the relations for the entities they extract. In these cases, precision and recall are as per the definition (equation 3.8, 3.9), and give a better idea of the automated systems performing the task from raw text. Other teams, on the other hand, consider corpus entities.

None of the systems identify the pair of entities needed to classify the relation.TempEval 2013, on the other hand, requires the automated systems to additionally do the task of identifying the pair of entities.

According to TimeML (PCI$^+$03) full-set relations, the temporal relation between two entities can be: *before, immediately before, includes, begins, ends*, and their inverses and *simultaneous*. On the other hand, according to TempEval shared tasks (VGS$^+$07), (VSCP10), the relation can be: *before, after, overlap, before-or-overlap, overlap-or-after* and *vague*. Besides exact matching, relaxed matching is also considered when considering the coarse-grained relations of TempEval. Precision and recall for relaxed matching are defined as:

$$Relaxed\ Precision = \frac{R_{correct} w}{R_{sys}} \tag{3.10}$$

$$Relaxed\ Recall = \frac{R_{correct} w}{R_{ref}} \tag{3.11}$$

where, $R_{correct}w$ reflects the weighted number of correct answers, R_{sys} indicates the total number of answers in the system annotation, and R_{ref} indicates the total number of answers in the reference annotation. In the weighted scoring, a response is not simply counted as 1 (correct) or 0 (incorrect), but it is assigned a specific value as shown in Table 3.10.

	before	*overlap*	*after*	*before-or-overlap*	*overlap-or-after*	*vague*
before	1	0	0	0.5	0	0.33
overlap	0	1	0	0.5	0.5	0.33
after	0	0	1	0	0.5	0.33
before-or-overlap	0.5	0.5	0	1	0.5	0.67
overlap-or-after	0	0.5	0.5	0.5	1	0.67
vague	0.33	0.33	0.33	0.67	0.67	1

Table 3.10: Relation Evaluation weights

Our implemented system is a corpus-based system. As a result, the concepts discussed in this chapter are used extensively in our proposed systems (Chapter 4).

3.4 Summary

We describe the existing computational linguistics approaches for temporal information understanding. We start the chapter by citing some benefits of *corpus linguistics* (MW96). In the first part of the chapter, we briefly describe the annotation schemes for *event, temporal expressions* and *temporal relations*, mostly in terms of TimeML (PCI$^+$03).

Next, we summarize the existing approaches for the extraction tasks. We find that the top-performing systems are approached in both *rule-based* and *data-driven* way for *event extraction.* The performance depends mainly on how much linguistic information is being used by the systems. Similarly, for *temporal expression extraction* and *temporal relation classification*, both *rule-based* and *data-driven* approaches achieve top performances.

As suggested by TempEval shared tasks (VGS$^+$07), (VSCP10), *temporal relation classification* was evaluated mostly in coarse-grained relations: *before, after, overlap, before-or-overlap, overlap-or-after* and *vague*, instead of using a full-set of TimeML relations. The existing systems also did not identify which pair of entities to consider for temporal relations. They assumed the pair of entities from the human annotated corpus, and then classified the temporal relations. However, in the upcoming temporal evaluation shared task – TempEval 2013, the participants first need to identify the pair of entities which has a temporal relation, and then need to classify the temporal relation into one of the TimeML relations[11].

We conclude the chapter by discussing the existing evaluation metrics for temporal information extraction tasks.

[11] *before, immediately before, includes, begins, ends* and their inverses, and *simultaneous.*

4 Temporal Information Extraction

In this chapter, we describe our approach for temporal information processing, which includes extracting **events** (TimeML `EVENT` tag), **temporal expressions** (TimeML `TIMEX`) and **identifying temporal relations** (TimeML `TLINK` tag). Our approach for all the tasks is best described as a **hybrid between linguistically motivated solutions and machine learning classifiers**. We perform deep semantic parsing and use hand-coded rules to extract *events, features* and *temporal expressions* from the logical forms produced by the semantic parser. In parallel, we filter *events, extract event features, temporal expressions*, and classify *temporal relations* using machine-learning classifiers.

This chapter is structured as follows: the first section demonstrates the general architecture and the flowchart of our approaches. The second section describes our system modules. The following sections discuss our approach for automatically extracting *events*, *temporal expressions*, identifying *temporal relations* and building an *end-to-end system* from raw text.

4.1 The General Architecture

We propose two systems – TRIOS and TRIPS – for automatically extracting temporal information from text. TRIOS is a hybrid system and TRIPS's *event extraction* module is primarily rule-based. Figure 4.1 illustrates the architecture of TRIOS system showing its different components.

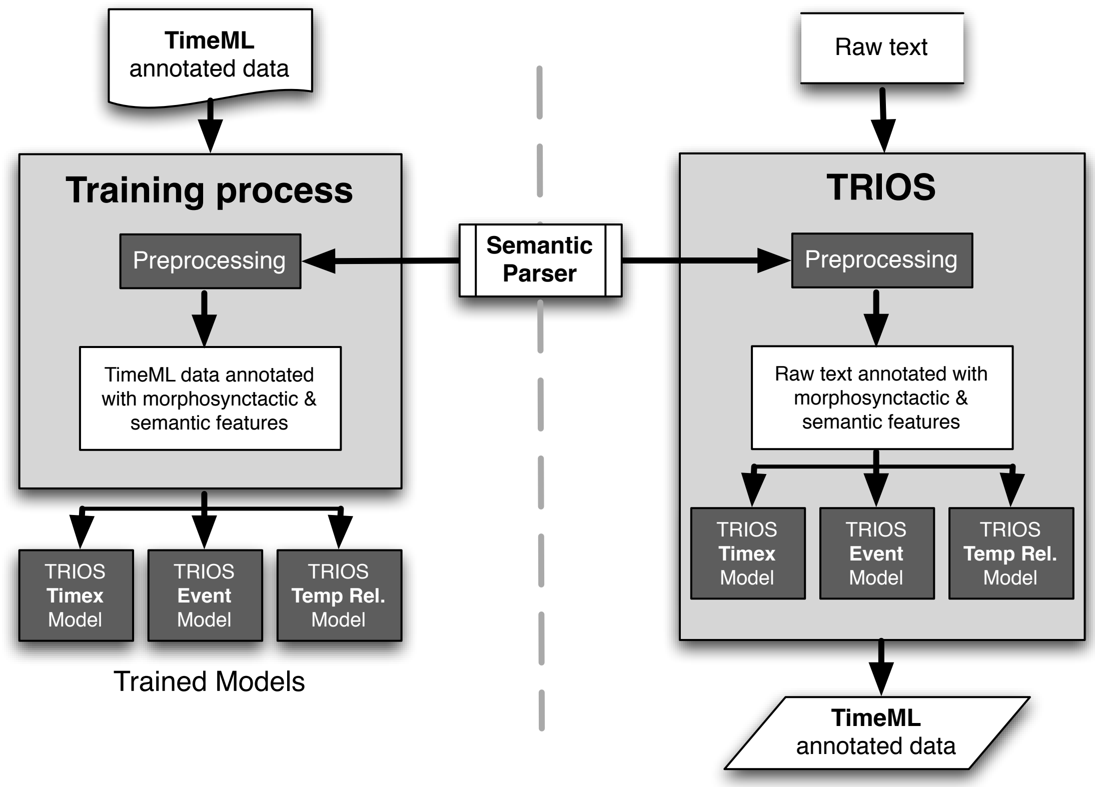

Figure 4.1: TRIOS architecture

Since TRIOS is a hybrid system with data-driven modules, it consists of two processes: the training process and the application process.

In both processes, the first step is the preprocessing of the input, where we obtain the *morphosyntactic features* and *semantic features (lexical semantics* *and sentence-level semantic features)*. Our semantic parser (section 4.2.1) primarily produces these features. We also have other classifiers to get event attributes for TRIOS system. Our detailed flowchart for both TRIOS and TRIPS systems is shown in Figure 4.2. TRIPS system output is color coded to blue; TRIOS system output is color coded to red; and the common output (shared by both TRIPS and TRIOS systems) is color coded to purple.

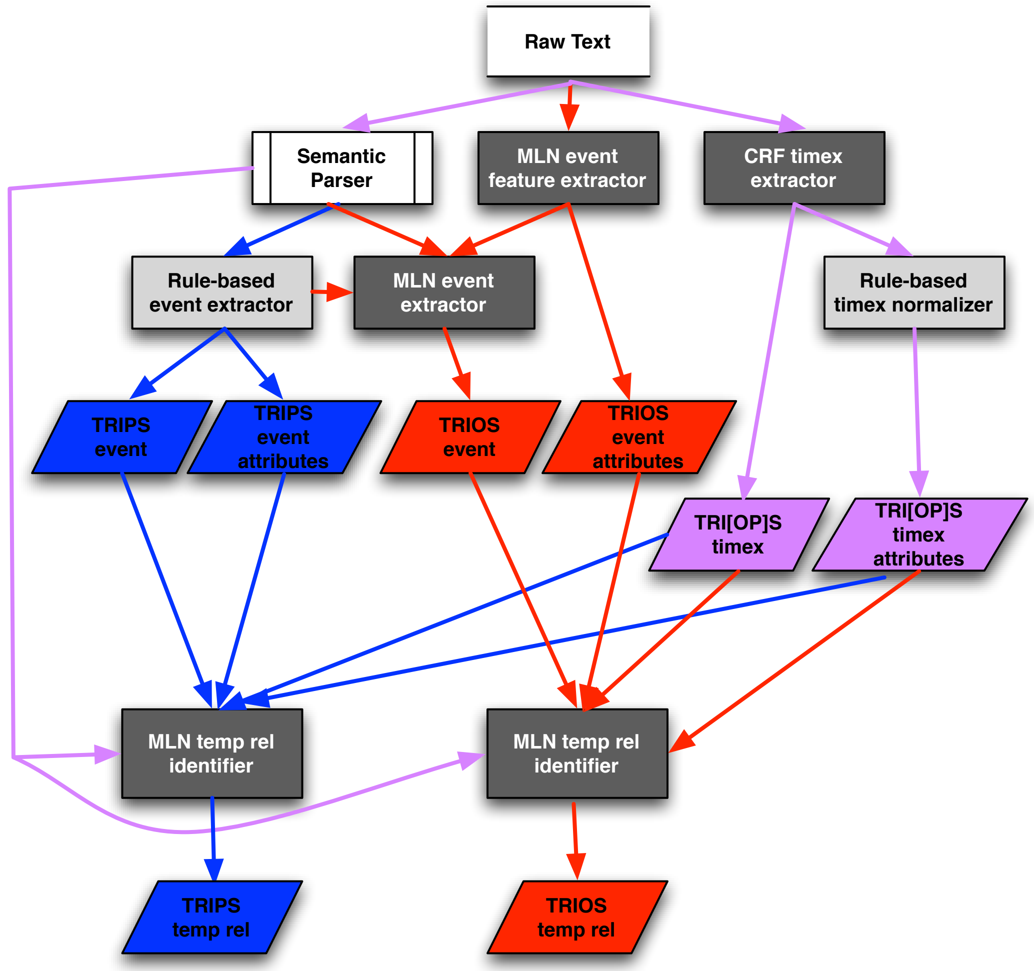

Figure 4.2: TRIOS/TRIPS flowchart

4.2 System Modules

In this section we briefly describe our system modules, and in the next sections we discuss how these modules are used in solving different subtasks.

4.2.1 TRIPS Semantic Parser

We use the existing TRIPS semantic parser (Allen et al., 2008) (ASB08) to produce deep logical forms from text. The system is generic and no grammatical rules or lexical entries were added specifically for this task. The TRIPS grammar is lexicalized context-free grammar, augmented with feature structures and feature unification. The grammar is motivated from X-bar theory, and draws on principles from GPSG (e.g., head and foot features) and HPSG. The parser uses a packed-forest chart representation and builds constituents bottom-up using a best-first search strategy similar to A*, based on rule and lexical weights and the influences of the statistical preprocessing. The search terminates when a pre-specified number of spanning constituents have been found or a pre-specified maximum chart size is reached. The chart is then searched using a dynamic programming algorithm to find the least cost sequence of logical forms according to a cost table that can be varied by genre.

The TRIPS system here uses a wide range of statistically driven preprocessing, including part of speech tagging, constituent bracketing, interpretation of unknown words using WordNet, and named-entity recognition (Allen et al., 2008) (ASB08). All these are generic off-the-shelf resources that extend and help guide the deep parsing process.

The TRIPS LF (logical form) ontology is designed to be linguistically motivated and domain independent. The semantic types and selectional restrictions are driven by linguistic considerations rather than requirements from reasoning components in the system (Dzikovska et al., 2003 (DAS03)). As much as possible the semantic types in the LF ontology are compatible with types found in FrameNet (Johnson and Fillmore, 2000 (JF00)). FrameNet generally provides a good level of abstraction for applications since the frames are derived from corpus examples and can be reliably distinguished by human annotators. However, TRIPS parser uses a smaller, more general set of semantic roles for linking the syntactic and semantic arguments rather than FrameNet's extensive set of specialized frame elements. The LF ontology defines approximately 3000 semantic types and 30 semantic roles. The TRIPS parser will produce LF representations in terms of this linguistically motivated ontology[1].

As an example, the result of parsing the sentence, *He fought in the war*, is expressed below as a set of expressions in an unscoped logical formalism with reified events and semantic roles.

(4.1).
```
(SPEECHACT V1 SA-TELL :CONTENT V2)
(F V2 (:* FIGHTING FIGHT) :AGENT V3 :MODS (V4) :TMA ((TENSE
PAST)))
(PRO V3 (:* PERSON HE) :CONTEXT-REL HE)
(F V4 (:* SITUATED-IN IN) :OF V2 :VAL V5)
(THE V5 (:* ACTION WAR))
```

The main event (`V2`) is of ontology type *fighting*, which is a subclass of *intentional-action*, and which corresponds to the first WordNet sense of *fight*, and includes verbs such as *fight, defend, contend and struggle.* The *agent* role of this event is the referent of the pronoun *he*, and the event is SITUATED-IN an event described by the word *war*. For words which are not in the TRIPS core lexicon, the system looks up the WordNet senses and maps them to the

[1]TRIPS ontology browser: `http://www.cs.rochester.edu/research/trips/lexicon/browse-ont-lex.html`

TRIPS ontology. For example, the word *war* is not in the core lexicon, and via WordNet it is classified into the TRIPS ontology as the abstract type *action*.

4.2.2 Markov Logic Networks (MLN)

Next we use a statistical relational learning (SRL) framework that recently gained momentum as a platform for global learning and inference in AI. It is Markov Logic (Richardson and Domingos, 2006 (RD06)). Markov logic is a combination of first order logic and Markov networks. It can be seen as a formalism that extends first-order logic to allow formulae to be violated with some penalty.

Formally, an MLN is a set of weighted first-order formulae. Given a set of constants, an MLN can be instantiated into a ground Markov network where each node is an atom. Each formula represents a feature in the grounded Markov network with the corresponding weight. The probability of an assignment x is

$$P(x) = \frac{1}{Z} exp^{\sum_i w_i n_i(x)} \tag{4.1}$$

where Z is the normalization constant, w_i is the weight of the i^{th} formula and $n_i(x)$ is the number of satisfied groundings for the i^{th} formula. MLN is a flexible way to incorporate human knowledge, since they allow using different combinations of features in a straight-forward manner by setting different formula templates and then learning the weights from the training data.

For our different classification tasks, we used different classifiers based on MLNs. We used an off-the-shelf MLN classifier Markov *thebeast*[2], using Cutting Plane Inference (Riedel, 2008) (Seb08) with an Integer Linear Programming (ILP) solver for inference.

[2]Markov *thebeast* is available online at: `http://code.google.com/p/thebeast/`

To use *thebeast* or any other MLN framework, at first we have to write the formulas, which corresponds to defining features for other machine learning algorithms. The Markov network then learns the weights for these formulas from the training corpus and uses these weights for inference in the testing phase.

An easy example below gives a brief idea about these weights. To classify the event feature *class*, we have a formula that captures the influence of both tense and aspect together. Here are two examples that show the learned weights for the formulas from training data[3].

```
tense(e1, INFINITIVE) & aspect(e1, NONE)
        => class(e1, OCCURRENCE) weight = 0.32

tense(e1, PRESPART) & aspect(e1, NONE)
        => class(e1, REPORTING) weight = -0.27
```

The MLN then uses these weights to reason about *class*. Generally, a more positive weight indicates that the formula is likely to hold. If the weight is negative then the formula most likely does not hold.

[3]There are many examples in our training data for *tense* INFINITIVE and *aspect* NONE together, which lead to the event *class* OCCURRENCE, e.g. *Wong Kwan will be lucky to **break** even. It is not going to **change** for a couple of years.* Hence, `tense(e1, INFINITIVE) & aspect(e1, NONE) => class(e1, OCCURRENCE)` formula has a higher positive weight. On the other hand, our training data does not include instances of *tense* PRESPART and *aspect* NONE together leading to the event *class* REPORTING, hence the formula `tense(e1, PRESPART) & aspect(e1, NONE) => class(e1, REPORTING)` has a negative weight, i.e. it is very unlikely to happen. One instance of *tense* PRESPART and *aspect* NONE from our training data is – *Initially, the company said it will close its commercial real-estate lending division, and stop **originating** new leases at its commercial lease subsidiary.* In these examples, the example events for which we are showing the *tense, aspect* and *class*, are shown in bold. PRESPART means present participle.

Finding useful features for MLNs is the same as finding them in any other machine learning algorithms. However, the MLN framework gives the opportunity to combine different features in the first order logic, which can lead to a better inference. For example, when filtering events, we have formulas that combine *word and pos*, or *word and previous word*, or *pos and next pos*, where we can capture the relationship between two predicates. Many of these predicates (features) could be encoded in other classifiers by concatenating the features. However, the increasing size a formula complicates matters. As a result, we have to regenerate the whole classifier data every time we introduce a new relationship.

4.3 The Event Extraction

For event extraction, we parse the raw text with the TRIPS parser. Then we take the resulting Logical Form (LF) and apply around hundred of hand-coded extraction patterns to extract events and features, by matching semantic patterns of phrases. These hand-coded rules are devised by checking the parse output in our development set. It was 2-3 weeks of work to come up with most of the extraction rules that extracts the events. There were minor incremental improvements in rules afterwards. It is worth mentioning, these rules are very generic and can be used in a new domain without any extra work, because, the TRIPS parser and ontology are domain independent, and use mappings from WordNet to interpret unknown words. Hence, the extraction rules will apply (and can be tested) for any natural language text without any extra work.

Because of the ontology, we can usually express general rules that capture a wide range of phenomena. For instance, all noun-phrases describing objects that fall under the TRIPS Ontology's top-level type *situation-root* are extracted as described events. This situation is captured by the extraction rule:

```
((THE ?x (? type SITUATION-ROOT))
     -extract-noms>
          (EVENT ?x (? type SITUATION-ROOT) :pos NOUN
                                    :class OCCURRENCE ))
```

By applying this formula in example (4.1), we can extract “war” as a nominal event, since “war” has the type *action*, which falls under *situation-root* in TRIPS ontology, this extraction rule will match the LF `(THE V5 (:* ACTION WAR))` and will extract *war* as event. Beside matching war under *situation-root* in ontology, it also matches the specifier *the*, which indicates that it is a definite noun phrase.

The result of matching around hundred of such rules to example (4.1) sentence is:

```
<EVENT eid=V2 word=FIGHT pos=VERBAL ont-type=FIGHTING
      tense=PAST class=OCCURRENCE  voice=ACTIVE aspect=NONE
      polarity=POSITIVE nf-morph=NONE>
<RLINK eventInstanceID=V2 ref-word=HE ref-ont-type=PERSON
      relType=AGENT>
<SLINK signal=IN eventInstanceID=V2 subordinatedEventInstance=V5
      relType=SITUATED-IN>
<EVENT eid=V5 word=WAR pos=NOUN ont-type=ACTION class= OCCURRENCE
      voice=ACTIVE polarity=POSITIVE aspect=NONE tense=NONE>
```

In this way, we extract events and TimeML-suggested event features (*class, tense, aspect, pos, polarity, modality*) for our TRIPS system. We also extract a few additional features such as ontology type (ont-type). TimeML tries to capture event information by a high-level attribute *class* or *pos*. The ontology type feature captures more fine-grained information about the event, but still much higher level than the words. The extraction rules also map our fine-grained types to the coarse-grained TimeML event class. We also extract relations between events (SLINK), whenever one event syntactically dominates the other, so it extracts more than TimeML's SLINKs and another new relation, relation between event and its arguments (RLINK). Details about these new additions can be found in section 7.1.

The TRIPS system extracts events in the temporally annotated corpus (TimeBank (PHS^{+}03), TempEval 1 (VGS^{+}07) or TempEval 2 (VSCP10)) with high recall. However, this high recall comes with the expense of precision. The reasons for lower precision include, (i) the fact that generic events are not coded as events in the corpus, (ii) errors in parsing and, (iii) legitimate events found by the parser but missed by corpus annotators. To remedy this problem, we introduced a MLN based filtering classifier, using the event features extracted from the TRIPS parser. The formulas in MLN for filtering were derived by linguistic intuition and by checking the errors in our development set. We devised around 30 MLN formulas.

There were two goals for this filtering step: (1) Eliminating events that result from errors in the parse, and (2) Removing event-classes, such as generics, that were not coded in the corpus. The second goal is needed to perform a meaningful evaluation on the existing temporally annotated corpus. The resulting system, including parsing, extraction, and post-filtering, is named as **TRIOS system**.

The TRIPS parser and extraction rules already give us event features along with events, which is reported in the results as the **TRIPS system**. To improve the performance, we implemented a MLN classifiers (**TRIOS system**) for the *class, tense, aspect and pos* features, using the features generated from the TRIPS parser plus lexical and syntactical features generated from the text using the Stanford POS tagger[4]. TRIOS system reports the *polarity* and *modality* performance of TRIPS system. The Table 4.1 gives a summary of features used to classify these event features.

Event attribute	Common features	Additional features
Pos	Event word, event penn tag, verb word sequence [a], verb pos sequence, previous word of verb sequence, next word, next pos	none
Tense		pos, polarity, modality, voice (active or passive)
Aspect		pos, polarity, modality, voice, pos+previous-pos, pos+next-pos
Class		TRIPS class suggestion, ont-type, slink-core-rel , tense+aspect, pos, stem, contains dollar

[a]One Penn tag derived features is verb word sequence, which captures all previous verbs, or TO (infinitive), or modal verbs, of the event word. That is, it will capture all consecutive verbs before the event until we get non-verbal word. Similarly verb pos sequence is the penn tag sequence of these verbs.

Table 4.1: Attributes/features used for classifying event features *pos, tense, aspect and class*

4.3.1 Evaluation and Discussion

Event Extraction: As mentioned earlier, the TimeBank corpus (Pustejuvsky et al., 2003 (PHS$^+$03)) is annotated according to TimeML (PCI$^+$03) specification. Later in TempEval (Temporal Evaluation shared task) (Verhagen et al., 2007 (VGS$^+$07)), the same corpus was released with modified event relations and minor modifications on some event features. Before describing our results and comparing with others, it is important to more carefully define the notion of *event* according to the TimeML specification.

[4]http://nlp.stanford.edu/software/tagger.shtml

TimeML considers *events* to be a cover term for situations that *happen* or *occur.* Events can be punctual or last for a period of time. They consider predicates describing *states* or *circumstances* in which something obtains or holds true. Events are generally expressed by means of tensed or untensed verbs, nominalizations, adjective, predicative clauses, or prepositional phrases. In addition, the TimeML specification says not to tag generic interpretations, even though capturing them could be of use in question answering. By generics, they mean events that are not positioned in time or in relation to other temporally located events in the document. For example, they won't annotate *use* and *travel* in the sentence: *Use of corporate jets for political travel is legal.*

In addition, subordinate verbs that express events which are clearly temporally located, but whose complements are generics, are not tagged, For example, *He said participants are prohibited from mocking one another.* Even though the verb said is temporally located, it isn't tagged because its complement, *participants are prohibited from mocking one another*, is generic.

And finally, event nominalization that do not provide any extra information than the supplied verb are also not tagged.

As for event attributes, TimeML considers class, tense, aspect, and nf-morph (Non-finite morphology). TimeBank contains 183 newswire documents. Later in the TempEval-1 (VGS$^+$07) contest, they use the same documents of TimeBank with some modification. One modification being removing the *nf-morph* attribute and introducing *pos* tag (part of speech) with VERB, ADJECTIVE, NOUN, PREPOSITION, OTHER. They modified the tense with PRESENT, NONE, PAST, FUTURE, INFINITIVE, PRESPART, PASTPART, to include rest of the values of *nf-morph.*

All our initial experiments are on TempEval-1 corpus and we also provide evaluation on TempEval-2. As a result, none of the existing systems contain performance of *pos* tag and our performance of the *tense* feature is also not comparable with other systems. However, for our rest of the experiments TimeBank and TempEval-1 corpus are same, so we will loosely refer to TempEval-1 as TimeBank when comparing with other systems.

The TempEval-1 corpus is divided into a training set of 163 documents and a test set of 20 documents. We used TempEval-1 test data as our development set for event extraction, and report the average of 10 cross-fold validation performance on the training data, which is totally unseen in our development.

System	Precision	Recall	Fscore	(P+R)/2
TRIPS avg	0.5863	**0.8422**	0.6914	0.7143
TRIOS avg	**0.8327**	0.7168	**0.7704**	0.7748
IAA	N/A	N/A	N/A	0.78

Table 4.2: Event Extraction Performance on TempEval Training data with 10 cross-fold validation (average)

Table 4.2 shows our performance on event extraction, where we also report Inter-annotator agreement (IAA[5]). The TRIPS system is the system with TRIPS parser and hand-coded extraction rules. We can see that TRIPS system gets a very high recall but with the expense of precision. As mentioned earlier, our errors result from tagging generic events and event that the TimeBank annotators missed, as well as true errors.

Bethard and Martin (2006) (BM06) (STEP system) had the prior state-of-the-art performance on event extraction in TimeBank corpus. They evaluated their system in 18 documents from TimeBank corpus and compared with other baselines. The EVITA (Sauri et al., 2005) (SKVP05) also implemented the

[5]Inter-annotator agreement (IAA) on subset of 10 documents from TimeBank 1.2. TempEval annotation for EVENT and TIMEX3 were taken verbatim from TimeBank 1.2 (Verhagen et al., 2007). IAA source: http://www.timeml.org/site/timebank/documentation-1.2.html#iaa

event extraction system on TimeBank corpus. However, their performance is inflated due to the fact that some aspects of their system were trained and tested on the same data. To get an idea of how well EVITA performs in an unseen data, Bethard and Martin simulated the EVITA system, which they called Sim-Evita. Another of their baselines is Memorize, which assigns to each word the label with which it occurred most frequently in the training data. To compare the performance of our systems, we tested in the STEP test set[6] and all the performances are reported below in Table 4.3 [7].

System	Precision	Recall	Fscore	(P+R)/2
TRIOS	**0.8638**	0.7074	**0.7778**	**0.7856**
TRIPS	0.5801	**0.8513**	0.6900	0.7157
STEP	0.82	0.706	0.7587	0.763
Sim-Evita	0.812	0.657	0.727	0.7345
Memorize	0.806	0.557	0.658	0.6815
IAA	N/A	N/A	N/A	0.78

Table 4.3: Event Extraction Performance on Bethard and Martin's test data

Again our TRIPS system has the highest Recall, which is around 15% higher than any other existing systems. But our TRIOS system outperforms any other systems in both precision and recall in TimeBank. STEP's recall is closest and almost similar to us, but we gain in MLN filtering step and end up with an overall higher precision. A 10-cross validation performance comparison

[6]Their testing documents are: APW19980219.0476, APW19980418.0210, NYT19980206.0466, PRI19980303.2000.2550, ea980120.1830.0071, and the wsj-XXXX documents numbered 0122, 0157, 0172, 0313, 0348, 0541, 0584, 0667, 0736, 0791, 0907, 0991 and 1033.

[7]Performances of STEP, Sim-Evita and Memorize is reported from Bethard and Martin (2006) (BM06)

for all systems would have given a better evaluation, but information is not available for the other systems.

We also participated in TempEval-2 (VSCP10) challenge. The TRIPS system has the highest recall in TempEval-2 too, while TRIOS system is second-best in precision with the highest scoring system, TIPSem. But overall TIPSem does very well compared to our system on event extraction. Performance of our both systems and the best performing TIPSem system is reported in Table 4.4.

System	Precision	Recall	Fscore
TRIOS	0.80	0.74	0.77
TRIPS	0.55	**0.88**	0.68
Best (TIPSem)	**0.81**	0.86	**0.84**

Table 4.4: Performance of Event Extraction (Task B) in TempEval-2

Event Extraction in New Domains: Our TRIPS system on event extraction is based on the domain independent semantic parser (TRIPS parser) and the domain independent extraction rules, hence porting to new domain is no extra work. To see how well TRIPS system performs in a new domain, we did an evaluation on two medical text documents (patient reports) with 146 events (human evaluated according to TimeML guideline (PCI$^+$03)) and found that TRIPS system performed similarly in a new domain as well. Our comparison is shown in Table 4.5.

This performance is suggestive that TRIPS system will have equivalent performance in new domains, but not conclusive, since it was tested in just two documents with 146 events on medical texts. On the other hand, the better TRIOS system is dependent on machine learning classifiers, which depends on having a training corpus. So, we cannot get equivalent performance of TRIOS

System	Precision	Recall	Fscore
TRIPS in TempEval-2	0.55	0.88	0.68
TRIPS in Medical Text	0.60	0.83	0.70

Table 4.5: Performance of TRIPS system in new (medical) domain vs TRIPS system in old (news) domain

system in new domains without the labeled training corpus.

Event Feature Extraction: Next, we discuss our performance on event feature extraction. Bethard and Martin (2006) (BM06) only report performance for event feature *class*; however, they report identifying class and event together in terms of precision and recall. This gives an idea of how accurately these features are extracted (precision) and from raw text how many events are extracted with correct class feature (recall). We compare these results directly with the TRIOS results in Table 4.6.

System	Precision	Recall	Fscore
STEP	0.667	0.512	0.579
TRIOS	**0.780**	**0.551**	**0.650**

Table 4.6: Event and Class Identification Performance on Bethard and Martin (2006)'s test set

Our main gain over the STEP system is in precision, and we also do better than them in recall. In addition to the general linguistically motivated features, our extracted *pos, tense, aspect and suggestions from TRIPS system* are used for identifying the class, which improves our performance. There are also two other systems that report the performance of class identification on TimeBank. They are EVITA (Sauri et al., 2005 (SKVP05)) and Chambers et al. (2007) (CWJ07), but they evaluate the accuracy ratio, i.e. the percentage of values their system marked correct according to the gold standard. For identifying class, EVITA assigns the class for the event that was most frequently assigned to them in the TimeBank. As before, this evaluation is trained and tested on the same document. With this technique they got an accuracy of 86.26%. Chambers et al. (2007) also had their majority class baseline, which is same as EVITA, except it does not train and test on the same document. The majority class baseline performance is 54.21%, a better estimate of EVITA's performance on class identification. The remaining three features that we extract are *pos, tense* and *aspect*. As mentioned in the beginning of this section, our experiments are on TempEval-1 corpus, which has different *tense* values than TimeBank, our performance on *tense* is not directly comparable. TimeBank also did not have *pos* feature. However, the performance of aspect can be compared with other systems. We are still reporting tense performance of other systems and inter-annotator agreement in all cases. Along with the accuracy (precision) numbers, we will also report the recall, which means what percentage of instances we extracted the event and got these features right, i.e. it is strictly dependent on event extraction's accuracy and always lower than that[8]. Our output is gathered from a 10-fold cross validation on the TempEval-1 training data.

EVITA outperforms us, with very small margin, in identification of aspect and tense, but it is important to recall that we are identifying both nf-morph and tense in the tense feature. EVITA's performance on nf-morph identification is 89.95%. This means, both systems perform almost equally well in this task. In pos, our performance is dependent on the third-party pos-tagger software. However, a naive baseline method that generates the TimeBank pos tags from tagger output has an accuracy of around 87%. Finally, in identifying class, we do significantly better than any other existing systems.

[8]Check section 3.3.1 for detailed explanation of the evaluation metrics.

System	Precision or Accuracy				Recall
Feature	TRIOS 10cv	Chambers'07 10cv	EVITA	IAA	TRIOS 10cv
Class	**0.8025**	0.752	0.5421	0.77	0.5749
Tense	0.9105	0.8828[1]	0.9205[1]	0.93	0.6523
Aspect	0.9732	0.9424	**0.9787**	1	0.6973
Pos	**0.9414**	N/A	N/A	0.99	0.6743

[1]Not directly comparable because their corpus had different values for *tense*

Table 4.7: Accuracy or Precision of Event Features and Recall of Event and Event Feature extraction (TRIOS on 10 cv on TempEval training data)

On TempEval-2 event feature extraction, our MLN-based TRIOS system has the best performance on aspect and polarity; we also do very well (second-best performances mostly) on *tense, class, pos and modality.* Performance of our systems' (TRIPS system's features are generated by TRIPS parser) event feature extraction in TempEval-2, along with the best team's performance is reported in Table 4.8

System	TRIPS	TRIOS	Best
Class	0.67	0.77	0.79 (TIPSem)
Tense	0.67	0.91	0.92 (Edinburgh-LTG)
Aspect	0.97	**0.98**	0.98
Pos	0.88	0.96	0.97 (TIPSem, Edinburgh-LTG)
Polarity	**0.99**	**0.99**	0.99
Modality	0.95	0.95	0.99 (Edinburgh-LTG)

Table 4.8: Performance of Event Features on TempEval-2 (Task B)

4.3.2 Main event identification

The *main event* represents the most important event in the sentence. An example will help to understand the *main event* better. For the sentence, *"Also today, King Hussein of Jordan **arrived** in Washington seeking to mediate the Persian Gulf crisis."*, we have four events: *arrived, seeking, mediate* and *crisis*. But ***arrived*** is the main event here. To build a temporally aware system, we need to identify the temporal relations between main events of the consecutive sentences.

We approach the problem of identifying the *main events*, given all the events, as another classification problem. We take our extracted events from the previous step and run a Markov Logic Network classifier[9] to classify the *main events* of a sentence.

In one of the tasks for TempEval 2010 (VSCP10), *main events* were labeled. We used that labeled data to train our *main event classifier*. As features, we used *lexical features* (word, stem, next word, previous word, previous verbal word sequence), *morphosyntactic features* (part-of-speech (pos) tag, tense, voice, polarity, TimeML aspect, modality, pos sequence, previous verbal pos sequence, next pos, previous pos), *lexical semantic features* (abstract semantic class ontology type) and *sentence-level semantic features* (TimeML class, semantic roles and their arguments) of the events. The *syntactic* and *semantic* features are mostly generated from the TRIPS parser (details in section 4.2.1) output and also using other classifiers.

The classifier first identifies the main events from the sentences. Then we run another pass to ensure every sentence has at least one *main event*. We force every sentence to have a main event. If a classifier did not identify a main event in a sentence, then we consider the first *verbal event* as the main

[9]Check section 4.2.2 for Markov Logic Network

event of the sentence. We show in the Evaluation that this model is a good baseline and adding this back-off model improves the performance significantly as well. We also show that with very naive features such as the lexical features, the part-of-speech tag derived features and the back-off model, we can get a high performing system, which performs better than a classifier trained on the TimeML-defined event features. However, we get the best performance by including the semantic features.

In the TempEval-2 data we have found that approximately 1.5% of the sentences have more than one main event. This might not be a very significant number, but these instances are important and significant in many domains. They represent conjunctions or colon separated sentences. Our framework handles these sentences because we do not have any constraint that there could be at most one main event in a sentence. Our classifier considers semantic information, semantic roles and other features. If any event has features related to main events then the classifier will try to classify it as main event. As a result, we handle the conjunctions and colon-separated sentences without any extra work.

Evaluation: We trained our main event identification classifier on TempEval-2 training data and tested it with 10-fold cross validation. We have several baselines that we explain below.

1. First event baseline (FEB): Consider the first event of the sentence as the main event

2. First verbal event baseline (FVB): Consider the first verbal event of the sentence as the main event

3. Hybrid baseline (HYB): Consider the first verbal event as the main event; if the verbal event does not exist in the sentence then consider the first event as the main event

4. Lexical and Penn tag features without verb word sequence related features (LPV): Run the classifier with lexical features and penn tag, previous pos and next pos

5. Lexical and Penn POS tag features (LPF): Run the classifier with features from 4-LPV + verb word sequence[10], verb pos sequence, previous word verb sequence, previous pos verb sequence

6. TimeML features generated by TRIOS (TFT): Run the classifier with TimeML features generated by TRIOS: word, pos, class, tense, polarity, aspect and modality

7. TimeML features taken from corpus (TFC): Run the classifier with TimeML gold standard features: word, pos, class, tense, polarity, aspect and modality

8. All features (ALF): Run the classifier with all features – lexical, syntactic and semantic features

At first we report the performance of all these baselines in Table 4.9 (first three rows). The classifier based main event extractor does not force the constraint that each sentence should have a main event. Hence, after classification, we run a back-off model with our hybrid baseline, i.e. if our classifier does not find a main event for a sentence, then it considers the first verbal event as main event and if there are no verbal events then considers the first event as main event. We report the performance of our baselines with back-off hybrid model in last three rows of Table 4.9. All the features used in the classifiers

[10]Verb word sequence captures all previous verbs, or TO (infinitive), or modal verbs, of the event word. That is, it will capture all consecutive verbs before the event until we get non-verbal word. Similarly verb pos sequence is the penn tag sequence of these verbs.

are generated by our systems, except 7-TFC, where we used the corpus gold standard features.

	1-FEB	2-FVB	3-HYB	4-LPV	5-LPF	6-TFT	7-TFC	8-ALF
Precision	0.6169	0.6444	0.6485	0.6940	0.7421	0.7120	0.7020	0.7599
Recall	0.4708	0.5423	0.5520	0.6516	0.7013	0.5983	0.5983	0.7299
Fscore	0.5340	0.5890	0.5964	0.6721	0.7211	0.6502	0.6460	**0.7446**
With	Hybrid	Backoff						
Precision	X	X	X	0.6802	0.7164	0.6877	0.6807	0.7315
Recall	X	X	X	0.7900	0.8209	0.7534	0.7540	0.8340
Fscore	X	X	X	0.7310	0.7651	0.7190	0.7155	**0.7794**

Table 4.9: Performance of main event identification and comparison between baselines

We observed the following from the experimental results:

1. Our classifier with all features (lexical, syntactic and semantic) performed best, which is 15-20% improvement over the naive baselines (1-FEB, 2-FVB and 3-HYB).

2. Incorporating hybrid baseline (3-HYB) as back-off model improved performance (3-6%) for all systems in the Fscore.

3. Just the lexical features and pos tag related features (5-LPF) can produce a high performing main event extractor.

   (a) 5-LPF performs 1-2% less than the best performing system with semantic feature (9-ALF); however, if someone wants to extract the main events without semantic computation then 5-LPF is a very good option.

(b) 5-LPF performs better than the system with TimeML attributes (class, tense, pos, aspect, modality, polarity), whether system generated (6-TFT) or from the gold standard (7-TFC).

(c) While just using the Penn tag derived features perform better than using the TimeML features, the verb word sequence related features made the difference. Comparison of 5-LPF and 4-LPV shows that.

4.4 Temporal Expression Extraction

4.4.1 Recognizing Temporal Expression

The TRIPS parser extracts temporal expressions the same way as we extract the events. The performance of the TRIPS parser's temporal extraction does not outperform state-of-the-art techniques on the evaluation measures. To improve on this, we also use a traditional machine learning classifier straight from the text. Our temporal expression extraction module is a hybrid between traditional machine learning classifier and the TRIPS parser extractor[11]. The TRIPS parser extracts some temporal expressions that are missed by our CRF based system and even sometimes missed by the TimeBank annotators. So we implemented a system by making a hybrid between CRF based system and TRIPS suggestion. The temporal expressions that are suggested by TRIPS parser but are missed by CRF based system, are passed to a filtering step that tries to extract a normalized value and type of the temporal expression. If we can find a normalized *value* and *type*, we accept these temporal expressions along with CRF based system's extracted temporal expressions.

For the machine learning classifier, we used a token-by-token classification for temporal expressions represented by B-I-O encoding with a set of lexical

[11]The system reported for TempEval-2 only uses the CRF based classifier.

and syntactic features, using Conditional Random Field classifier[12].

We used lexical features like *word, shape, is year, is date of week, is month, is number, is time, is day, is quarter, is punctuation, if belong to word-list like init-list*[13] *, follow-list*[14], etc. We then use CRF++ formulas in template to capture relation between different features to extract the sequence. For example, we will write a formula to capture the current word is in the *init-list* and the next word is in the *follow-list*, this rule will train the model to capture sequences like *this weekend, earlier morning, several years*, etc.

For TempEval-2, we did not make hybrid between CRF and TRIPS, but we still merge TRIPS and CRF, because, the TRIPS parser does extract temporal relations between events and temporal expressions, which helps us in the temporal relation identification tasks. So we take the temporal expressions from the CRF based extractor and for the cases where TRIPS parser extracts the temporal expression, we use TRIPS parser id, so that we can relate to relations generated by the parser.

4.4.2 Determining The Normalized Value and Type of temporal expression

Temporal expressions are most useful for later processing when a normalized *value* and *type* is determined. We implemented a rule-based technique to determine the *type* and *value*. We match regular expressions to identify the

[12]We used off the shelf CRF++ implementation. http://crfpp.sourceforge.net/

[13]*init-list* contains words like: *this, mid, first, almost, last, next, early, recent, earlier, beginning, nearly, few, following, several, around, the, less, than, more, no, of, each, late.*

[14]*follow-list* contains words like: *century, centuries, day, days, era, hour, hours, millisecond, minute, minutes, moment, month, months, night, nights, sec, second, time, week, weeks, year, years, am, pm, weekend, summer, fall, winter, fiscal, morning, evening, afternoon, noon, EST, GMT, PST, CST, ago, half.*

type of temporal expressions. *Type* could be either of *time, date, duration and set.*

Then in the next step we extract the normalized value of temporal expression, as suggested by TimeML scheme. We take the Document Creation Time (DCT) and then calculate the values for different dates in terms of document creation date, e.g. *last month, Sunday, today.* The *type* instances are trivial.

Our type and value extractor and temporal expression extractor modules are available[15] for public use.

4.4.3 Evaluation and Discussion

Recognizing Temporal Expressions: Our first evaluations are on TempEval-1 or TimeBank corpus and we compared our system against the systems that were developed before TempEval-2. Similar to event extraction, we used 10-fold cross validation on the TempEval corpus' training data to evaluate the performance of our system. Boguraev and Ando (BA-2005) (BA05), and Kolomiyets and Moens (KM-2009) (KM09) also report their performances on TimeBank. Tables 4.10 and 4.11[16] show the comparison between the existing systems and our two systems.

We can see that in the relaxed match we outperform existing systems before TempEval-2 and in strict match we do equally well with the best state-of-the-art system. However, our CRF with TRIPS system's performance did not outperform the CRF-alone system. To investigate, we hand-checked some of the suggestions of the TRIPS-based system on TempEval test set. We found that there are legitimate temporal expressions that were missed by the

[15]Details in Appendix A.

[16]Relaxed match admits recognition as long as there are any common words. Sloppy span admits recognition as long as right boundary is same in the corresponding TimeBank in-stance. Strict match admits recognition when both strings are strictly matched.

System	Precision	Recall	Fscore
KM-2009	0.872	0.836	0.852
BA-2005	0.852	0.952	0.896
CRF+TRIPS	0.8979	0.8951	0.8951
CRF	0.9541	0.8645	**0.9075**

Table 4.10: Temporal expression relaxed match extraction on TimeBank (BA-2005 uses sloppy span)

System	Precision	Recall	Fscore
KM-2009	0.866	0.796	**0.828**
BA-2005	0.766	0.861	0.817
CRF+TRIPS	0.8064	0.8038	0.8051
CRF	0.8649	0.7846	***0.8228***

Table 4.11: Temporal expression extraction strict match performance on TimeBank

TimeBank annotators. If we include those temporal expressions, then our TRIPS-based system outperforms any existing systems, including our CRF-based system. This experiment was conducted on only the TempEval test set. It is reported in the table 4.12 below.

System	Precision	Recall
CRF	0.8242	0.9069
CRF+TRIPS	0.8195	0.8990
CRF+TRIPS hand-verification	**0.8501**	**0.9296**

Table 4.12: Improvement of TRIPS+CRF based system over CRF based system show on TempEval-1 test set

We have shown that the CRF-based system with our selected lexical and grammatical features outperforms or does equally well with the existing systems. In addition, our TRIPS-based system performs the best when omissions from the Timebank corpus are taken into account. The result of TRIPS based system is suggestive, but not conclusive. We prefer to use this system in our future work on identifying temporal relations because TRIPS based system is backed by a domain independent semantic parser.

Determining Normalized Value and Type of temporal expressions: Most previous work on temporal expression extraction on the TimeBank corpus (Boguraev and Ando, 2005 and Kolomiyets and Moens, 2009) focused on just recognizing temporal expressions. Boguraev and Ando also report their performance on identifying *type*. We will show the comparison with Boguraev and Ando (BA-2005) on identifying *type*. Previous work before TempEval-2 did not report results on computing normalized values.

We considered the temporal expressions that are matched with the relaxed match and for these instances we checked the number of cases we identified the type and value accurately. Our 10-fold cross validation performance for both of our systems and performance of BA-2005 on Time-Bank is reported in Table 4.13, which shows we outperform Boguraev and Ando (2005).

System	*type* accuracy	*value* accuracy
CRF+TRIPS	0.906	0.7576
CRF	0.9037	0.7594
BA-2005	0.815	N/A

Table 4.13: Performance of type and value identification on TempEval for recognized (relaxed) temporal expressions

TempEval-2: On TempEval-2 (VSCP10), both the TRIPS and TRIOS systems used the same CRF based approach and we did not make any hybrid with TRIPS parser extraction. Our system attained the second best performance on combined temporal expression extraction and normalization task (identifying *type* and *value*). It is worth mentioning that the average of identifying *value* performance is 0.61 and if we remove our systems and the best system, HeidelTime-1, the average is only 0.56. Hence, our freely distributed normalization tool could be beneficial to many people. Performance of our system and the best system on task A is reported in Table 4.14.

4.5 Temporal Relation Identification

We identify the temporal relations using a Markov Logic Network classifier, namely *thebeast*, by using linguistically motivated features that we extracted in previous steps. Our work matches closely with the work of Yoshikawa et al. (2009) (YRAM09). We only consider the local classifiers, but we use more

		TRIPS TRIOS	Best HeidelTime-1
Temp Exp extraction	Precision	0.85	0.90
	Recall	0.85	0.82
	Fscore	0.85	0.86
Normalization	*type*	0.94	0.96
	value	0.76	0.85

Table 4.14: Performance on Temporal Expression extraction (Task A)

linguistically motivated features and features generated from text, whereas they use TempEval-1's (Verhagen et al., 2007 (VGS$^+$07)) annotations as input, along with their derived features.

TempEval-1 (VGS$^+$07) has three tasks on identifying temporal relations and TempEval-2 (VSCP10) has four tasks, which includes all TempEval-1's tasks and an additional task. We will describe our system and report our performance in terms of these tasks to compare our system with other competing systems.

The four subtasks for identifying temporal relations in TempEval-1 (TE1) and TempEval-2 (TE2) are (numbered according to both TempEval-1 and TempEval-2):
(TE1 A or TE2 C) Determine the temporal relation between an event and temporal expression in the same sentence;
(TE1 B or TE2 D) Determine the temporal relation between an event and the document creation time (DCT);
(TE1 C or TE2 E) Determine the temporal relation between the main events in two adjacent sentences; and
(TE2 F) Determine the temporal relation between two events, where one event

syntactically dominates the other event.

The Table 4.15 shows the features we used for each of these tasks. We used some features that we extracted from the TRIPS parser. Related information about these concepts can be found in UzZaman and Allen (2010) and also in later section 7.1 (TRIOS-TimeBank).

4.5.1 Evaluation and Discussion

In this section, we evaluate our system initially on TempEval-1 (VGS^{+}07) data and compare our system with the systems reported before TempEval-2. Next, we evaluate on TempEval-2 data comparing with the TempEval-2 participants.

As mentioned already, TempEval-1 contains the same documents as TimeBank. They divided the TimeBank into 163 documents for TempEval training data and 20 documents for TempEval-1 test data. They modified the temporal relation values in TempEval-1 and made it simpler.

In event and temporal expression extraction, we used the TempEval-1 test set as our development set and report the performance using 10-fold cross validation on TempEval-1 training set. In case of temporal relation identification, we do not actually use the TempEval-1 test set as our development set. But our extracted features were part of development set, so we report 10-fold cross validation on TempEval-1 training data as well to better understand the capability of our system. We also report the performance on TempEval-1 test data to compare with other systems.

Most of the systems in the TempEval-1 task used the features, events, and temporal expressions provided in the TempEval-1 corpus as input. In contrast, we extracted all the events, temporal expression and features from raw text and used our extracted information to identify the temporal relations. Two systems, LCC-TE (Min et al., 2007 (MSF07)) and XRCE-T (Hagege

Features	TE2 C	TE2 D	TE2 E	TE2 F
	TE1 A	TE1 B	TE1 C	
Event Class	YES	YES	e_1 x e_2	e_1 x e_2 [a]
Event Tense	YES	YES	e_1 x e_2	e_1 x e_2
Event Aspect	YES	YES	e_1 x e_2	e_1 x e_2
Event Polarity	YES	YES	e_1 x e_2	e_1 x e_2
Event Stem	YES	YES	e_1 x e_2	e_1 x e_2
Event Word	YES	YES	YES	YES
Event Constituent [b]		YES	e_1 x e_2	e_1 x e_2
Event Ont-type [c]	YES	YES	e_1 x e_2	e_1 x e_2
Event LexAspect [d] *x Tense*	YES	YES	e_1 x e_2	e_1 x e_2
Event Pos	YES	YES	e_1 x e_2	e_1 x e_2
Timex Word		YES		
Timex Type	YES	YES		
Timex Value	YES	YES		
Timex DCT relation	YES	YES		
Event's semantic role [e]	YES	YES		
Event's argument's ont-type	YES	YES		
TLINK event-time signal [f]	YES	YES		
SLINK event-event relation type [g]				YES

[a]In the MLN framework, we can write formulae in first-order logic. e_1 x e_2 instances are cases, where we capture both events together. For example, in case of Tense, it will learn the weights for temporal relations given first event's tense is PRESENT and second event's tense is PAST. Instead of just considering first event is PRESENT and second event is PAST, we are considering first event is PRESENT and second event is PAST together.

[b]TRIPS parser identifies the event constituent along with event word.

[c]Ontology-type is described in section 7.1.2.

[d]LexicalAspect classifies the events into *Event, State and Reporting* class.

[e]Semantic or thematic role that connects the event and its argument

[f]TRIPS parser generated event-time TLINK connective or signal (similar to TimeML)

[g]The SLINK relation type that connects two events, details in section 7.1.4.

Table 4.15: Features used for TempEval-2 (TE2) Task C, D, E and F or TempEval-1 (TE1) Task A, B and C.

and Tannier, 2007 (HT07)) extracted features from text. But LCC-TE has a component called Temporal Merger, which compares the event and temporal tagging and if there are inconsistencies then it matches with TempEval-1, which is to say for missing cases they will use TempEval-1 data. As a result, their recall is not actual recall and their precision is also slightly inflated by some TempEval-1 gold annotations. So we only compare with XRCE-T. However, XRCE-T's Task C's (event-event relations) recall is not comparable either. In cases where they missed an event, they use the default value OVERLAP, which is the most frequent relationship in Task C. So they get the same recall as precision, which does not represent the actual recall of Task C, i.e. how many events they correctly extracted from text and also identified the relations correctly.

In the Tables 4.16, 4.17 and 4.18, we report our performance (TRIOS) and compare with other systems. 10cv is 10-fold cross validation performance on the training data, Test is performance on test data and XRCE-T is the XRCE-T team's performance on test data. We also show the best and average performance for TempEval 1 participants.

	Strict			Relax		
	P	R	F	P	R	F
TRIOS on 10cv	0.58	0.42	**0.49**	0.61	0.45	**0.52**
TRIOS on Test data	0.56	0.43	**0.49**	0.58	0.45	**0.51**
XRCE-T on Test data	0.53	0.25	0.34	0.63	0.30	0.41
Best on TempEval-1	0.62	N/A	N/A	0.64	N/A	N/A
Avg on TempEval-1	0.59	N/A	N/A	0.62	N/A	N/A

Table 4.16: Performance on TempEval-1 Task A

We can see that our performance is competitive with the best and average of

	Strict			Relax		
	P	R	F	P	R	F
TRIOS on 10cv	0.75	0.65	**0.69**	0.79	0.67	**0.72**
TRIOS on Test data	0.76	0.67	**0.71**	0.78	0.69	**0.73**
XRCE-T on Test data	0.78	0.57	0.66	0.84	0.62	0.71
Best on TempEval-1	0.80	N/A	N/A	0.84	N/A	N/A
Avg on TempEval-1	0.76	N/A	N/A	0.78	N/A	N/A

Table 4.17: Performance on TempEval-1 Task B

	Strict			Relax		
	P	R	F	P	R	F
TRIOS on 10cv	**0.53**	0.41	**0.46**	**0.61**	0.47	**0.53**
TRIOS on Test data	**0.55**	0.42	**0.48**	**0.61**	0.47	**0.53**
XRCE-T on Test data	0.42	N/A [1]	0.66	0.58	N/A	N/A
Best on TempEval-1	0.62	N/A	N/A	0.64	N/A	N/A
Avg on TempEval-1	0.59	N/A	N/A	0.62	N/A	N/A

[1] XRCE-T assumed the events from TimeBank for task C. For events that they couldn't extract they assumed the default relation OVERLAP, which is the most frequent relations between consecutive sentences' main events.

Table 4.18: Performance on TempEval-1 Task C

the TempEval-1's performance, even though we are using our own extracted features and they are using the TempEval-1 annotations. Comparing with XRCE-T, the only system that extracts features from text like us, we have shown that in all tasks we outperform them in F-score. Our feature extraction and event and temporal expression extraction is mostly similar with XRCE-T, as both of the systems use semantic parsing. We eventually outperform them because our final system is a hybrid with machine learning classifiers, getting the best of both worlds.

On TempEval-2 (VSCP10), we submitted two systems, TRIPS and TRIOS. Both TRIPS and TRIOS use the same MLN classifier with same feature-set for each task. However the difference is, they take events and temporal expressions from their respective systems, e.g. in Task C (temporal relation between events and temporal expressions), TRIPS system will classify relations for instances where TRIPS event extractor extracted events (TempEval-2 task B) and TRIPS temporal expression extractor extracted temporal expressions (TempEval-2 task A). The recall measure of task C will represent the accuracy of extracting events, temporal expression and identifying temporal relations together. This applies for all C - F tasks and for our both TRIOS and TRIPS systems.

For temporal relation identification (Task C - F), most of the teams used events, temporal expressions and their features from the human-annotated corpus, whereas, we used our extracted entities and their features that we extracted in Task A and B. Hence, our performance represents the capability of identifying these relations from raw text and is a harder classification task, since we are starting with imperfect features.

Even though we are using our own generated features, we outperformed other groups in task C (temporal relation between events and temporal expressions) and task E (the relation between main events of consecutive sentences).

We also have second-best/equivalent performance for two other tasks (relation between event and DCT; and relation between events where one syntactically dominates other).

Table 4.19 reports our systems' performance with precision and recall. For others, since they take annotated events and features, they do not actually have a recall, hence their recall is not reported.

Since the TRIPS system for temporal relations (first two columns in Table 4.19) uses the TRIPS events (task B) (performance in Table 4.4), which has a higher recall, the TRIPS system has higher recall in relations as well.

	TRIPS		TRIOS		Best (with corpus features)
Task	Precision	Recall	Precision	Recall	Precision
Task C	0.63	**0.52**	**0.65**	**0.52**	0.63 (JU-CSE, UCFD, NCSU-indi)
Task D	0.76	**0.69**	0.79	0.67	**0.82** (TIPSem)
Task E	**0.58**	**0.50**	0.56	0.42	0.55 (TIPSem)
Task F	0.59	**0.54**	0.6	0.46	**0.66** (NCSU-individual)

Table 4.19: Performance of Temporal Relations on TempEval-2 (Task C-F)

4.6 An End-to-End System

In the previous sections, we presented our system that extracts temporal information from texts. However, we are still missing a few components required to build an end-to-end system, which would output the same information as TimeML annotations. To do that, we need to: (i) add a module to identify paired entities which have temporal relations between them; and (ii) train the classification model on the full set of TimeML relations instead of on the reduced set of TempEval-2 relations. With these additions, we can achieve an end-to-end system that creates TimeML annotations from raw text. We have implemented components to do these additional tasks.

For classification of the full set of temporal relations, we train on the TimeBank (PHS+03) and AQUAINT[17] corpora[18], instead of the TempEval-1 (VGS+07) or TempEval-2 data.

In order to meet the TimeML specifications, we only identify temporal relations between the *main events* of consecutive sentences, and also the relations between *events* & *temporal expressions* that are syntactic arguments of the *main events*. Our *main event* identification module is described in section 4.3.2.

The performance of our end-to-end system, along with the performance of the state-of-the-art system, is reported at the end of section 6.1.8.

4.7 Summary

We present our work on *extracting temporal information from raw text*. Our system for **extracting temporal information** uses a combination of deep semantic parsing and machine learning classifiers. We compare our system with the existing systems doing the same task on TimeBank (PHS+03) and TempEval (VGS+07), (VSCP10) corpora. Our system outperforms or performs comparably with the existing systems. Most importantly, our system performs all these tasks simultaneously within a single unifying framework. In contrast, most of the systems, which were compared to our system, were trained specifically to do just a single task. The performance of our unified system is best understood in the performance of temporal relation classification.

[17]http://timeml.org/site/timebank/timebank.html

[18]Cleaned and updated version of the corpus is available here: `http://www.cs.rochester.edu/~naushad/tempeval3/data/TBAQ-cleaned.zip`

In TempEval-2, our system outperforms other systems in the two temporal relation classification tasks and does equally well in the other two tasks by using our system generated features. Many systems, on the other hand, use gold corpus features instead of using self-generated features.

5 Temporal Question Answering

In this chapter, we present our temporal QA system[1], which performs temporal reasoning to infer implicit relations, and answers *temporal yes/no* and *list* questions.

The initial motivation of the temporal annotation scheme, TimeML (PCI$^+$03), was to tackle the complex temporal question answering (QA) task. Due to the complexity of temporal QA, the focus had been shifted to solve smaller subtasks, such as *extracting events, temporal expressions* or *identifying temporal relations* (VGS$^+$07), (VSCP10). As a result, the vast amount of literature on temporal reasoning[2] has not been widely applied in the corpus based temporal information processing research. We explain temporal reasoning in the following way: *"Given a set of explicit temporal relations between a set of entities, temporal reasoning infers additional relations between entities that is implicit in the ones given".*

We start the chapter by describing the existing temporal QA systems from corpus linguistics literature. Next, we present the types of questions our system answers, and finally we demonstrate that our system uses *temporal reasoning*

[1]Our QA system takes a TimeML (PCI$^+$03) annotated document as input.

[2]Check section 2.7.1 for Temporal Reasoning systems

to answer temporal questions.

5.1 The Existing Temporal Question-Answering Systems

There are some related work on temporal QA by doing temporal reasoning. Below we briefly describe these systems and their limitations.

Pustejovsky et al. (PWM02) discuss how TimeML could be used for temporal QA, but do not present any system to do so. Hobbs and Pustejovsky (HP03) also discuss using TimeML for temporal reasoning, and point out its importance for temporal QA. However, they do not present any system to do the task either.

Harabagiu and Bejan's work (HB05) on QA with temporal inference handles question-answering based on general inference but not temporal inference in pure sense. The following example briefly explains their approach. Given the question Q1: "When did Iraq invade Kuwait?" The answer to Q1 – "2 August 1990" is extracted from the context: "Iraqis have been struggling under UN sanctions ever since Hussein's annexation of Kuwait on 2 August 1990." Harabagiu and Bejan identify temporal relations between entities with temporal signals and then make semantic inference to match the question with the answer context, e.g. matching Iraq's invasion with Hussein's annexation. However, they do not infer implicit temporal relations with temporal reasoning.

Moldovan et al.'s work (MCH05) on temporal reasoning approach the problem very similarly as Harabagiu and Bejan. They identify the temporal relations between entities and feed a module in their overall inference system. They do inference between only temporal expressions. Both of these systems

((HB05) and (MCH05)) are unable to do complete temporal reasoning or answer questions such as: *listing what happened after some event* or *inferring when some event occurred with respect to other events*, etc.

Chambers and Jurafsky (CJ08) use transitive closure properties to consider the global inference to identify temporal relations between two entities. In the process they can identify how two entities are temporally related, if they are connected. However, their reasoning is limited to only *before, after* and *vague* relations, whereas, our system handles all 13 Allen relations.

The mainstream question answering systems are web-based systems and they rely heavily on answer redundancy instead of reasoning (BLB^{+}01), (SVMB^{+}09).

None of these QA systems can answer *temporal list* questions or perform the *temporal reasoning* as we defined in the beginning of this chapter. After few years of active research in temporal information processing, we currently have available corpora and automated extraction systems to develop an end-to-end advanced temporal QA systems performing temporal reasoning. In this section we report one such system.

The mainstream QA forum TREC[3] evaluated systems mainly on *factoid questions, list questions* and *other questions.* Like TREC, we consider *factoid*, and *list questions.* However, given our focus on temporal reasoning, we include a *yes-no question* category as well.

5.2 Temporal Question Taxonomy

We implemented a temporal QA system to address the following question types:

[3]http://trec.nist.gov/

1. (a) *yes/no*: "Was Fein called after the killings?"
   (b) *list*: "What happened after the crash?", "What happened between the crash and yesterday?"
   (c) *when (factoid)*: "When did DT Inc. holders adopt a shareholder-rights plan?"

To answer these questions the system has to be aware about the temporal relations of events and temporal expressions, which can be explicit in the TimeML annotation or implicit (inferred by temporal reasoning).

5.3 Building the Temporal Structure for Temporal Reasoning

We have to build the temporal structure (Figure 1.2) to infer how all entities are related to each other temporally. We can take benefit of existing temporal reasoning systems (for summary on temporal reasoning systems, check section 2.7.1) to do this task. We preferred Timegraph (Miller and Schubert, 1990) (MS90) over the widely used Allen's interval closure algorithm (Allen, 1983) (All83), because of Timegraph's scalablility[4] for larger problems (Yampratoom and Allen, 1993) (YA93). Furthermore, the additional expressive power of interval disjunction in Allen (1983) does not appear to play a significant role in the temporal extractions from text[5].

[4]Allen's temporal closure takes $O(n^2)$ space for n intervals, whereas Timegraph takes $O(n + e)$ space, where n is the number of time points and e is the number of relations between them. In terms of closure computation, without disjunction Allen's algorithm computes in $O(n^2)$, whereas Timegraph takes $O(n + e)$ time, n and e are same as before. Detail comparison about these two systems can be found in section 2.7.1.

[5]Allen's closure algorithm is computationally intractable if interval disjunctions are considered. However, Vilain, Kautz and van Beek (1986, 1990) (VK86), (VKvB90) showed that

In this section, we will describe the Timegraph algorithm and our Timegraph implementation for TimeML temporal relations.

5.3.1 Timegraph

A Timegraph G = (T, E) is an acyclic directed graph in which T is the set of vertices (nodes) and E is the set of edges (links). It is partitioned into chains, which are defined as sets of points in a linear order. Links between points in the same chain are in-chain links and links between points in different chains are cross-chain links. Each point has a numeric pseudo-time, which is arbitrary except that it maintains the ordering relationship between the points on the same chain. Chain and pseudo-time information are calculated when the point is first entered into the Timegraph. Determining relationship between any two points in the same chain can be done in constant time simply by comparing the pseudo-times, rather than following the in-chain links. On the other hand, relationship between points in different chains can be found with a search in cross-chain links, which is dependent on the number of edges (i.e. number of chains and number of cross-chain links). A *metagraph* keeps track of the cross-chain links effectively by maintaining a *metanode* for each chain, and using a cross-chain links between metanodes. As already described, in-chain checking can be done in constant time, and a graph search is dependent on the number of cross-chain links rather than the total number of points. Creation of all supporting graph structures (including the metagraph) requires $O(n+e)$ space and $O(n + e)$ time, where n is the number of time points, and e is the number of relations between them. More details about Timegraph can be found in Miller and Schubert (1990) (MS90) and Taugher (1983) (Tau83). Figure 5.1 shows an example of Timegraph.

Allen's closure algorithm is tractable if the interval disjunctions are not considered.

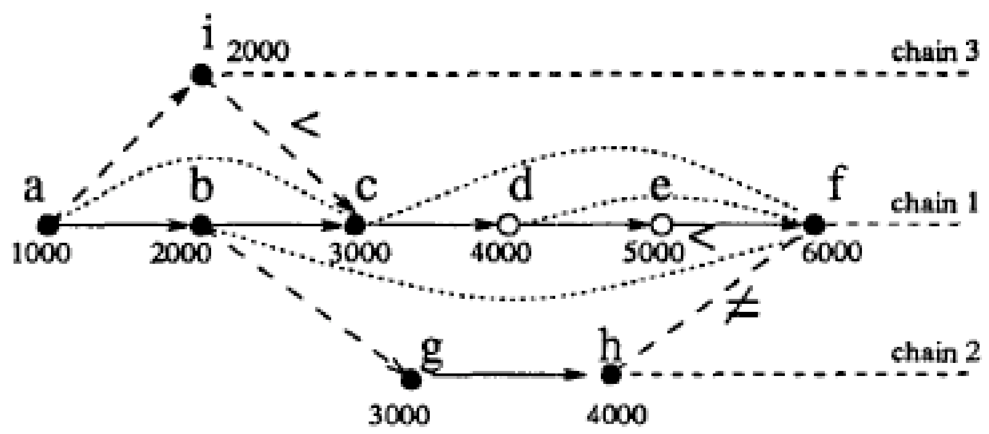

Figure 5.1: One Example of Timegraph

Timegraph only supports simple point relations (<, =, ≤), but we need to build the system based on TimeML relations, which is based on interval algebra. This is not a problem, since single (i.e., non-disjunctive) interval relations can be easily converted to point relations[6].

5.3.2 Timegraph for TimeML

We want to minimize the number of chains constructed by Timegraph to efficiently build Timegraph for TimeML relations. For each relation we have to make sure all constraints are met. The easiest and best way to approach this is to consider all relations together. For example, for interval relation X *includes* Y, the point relation constraints are: x1<y1, x1<y2, x2>y1, x2>y2, x1<x2 and y1<y2. We want to consider all constraints together as, x1 < y1 <

[6]Interval relation between two intervals X and Y is represented with points x1, x2, y1 and y2, where x1 and y1 are start points and x2 and y2 are end points of X and Y. Temporal relations between interval X and Y is represented with point relation between x1, y1; x1, y2; x2, y1 and x2, y2.

Allen relations	Equivalent in Point Algebra	Point Algebra represented as a chain
Before	x1<y1, x1<y2, x2<y1, x2<y2	x1 < x2 < y1 < y2
After	x1>y1, x1>y2, x2>y1, x2>y2	y1 < y2 < x1 < x2
Meet	x1<y1, x1<y2, x2=y1, x2<y2	x1 < x2 = y1 < y2
MetBy	x1>y1, x1=y2, x2>y1, x2>y2	y1 < y2 = x1 < x2
Start	x1=y1, x1<y2, x2>y1, x2<y2	x1 = y1 < x2 < y2
StartedBy	x1=y1, x1<y2, x2>y1, x2>y2	x1 = y1 < y2 < x2
Finish	x1>y1, x1<y2, x2>y1, x2=y2	y1 < x1 < x2 = y2
FinishedBy	x1<y1, x1<y2, x2>y1, x2=y2	x1 < y1 < y2 = x2
During	x1>y1, x1<y2, x2>y1, x2<y2	y1 < x1 < x2 < y2
Contains	x1<y1, x1<y2, x2>y1, x2>y2	x1 < y1 < y2 < x2
Equality	x1=y1, x1<y2, x2>y1, x2=y2	x1 = y1 < x2 = y2

Table 5.1: Interval algebra and equivalent point algebra

y2< x2 and add all together in the Timegraph. In Table 5.1, we show Allen's relations[7], equivalent representation in point algebra and finally point algebra represented as a chain, which makes adding relations in Timegraph much easier with fewer chains. These additions make Timegraph more effective for TimeML corpus.

In this way, we can create the Timegraph or the temporal structure for a TimeML document, which can answer how two entities are related with each other in terms of time. The following examples (Figure 5.2 to 5.5) demonstrates how we build an internal Timegraph representation from a TimeML annotation.

In Figure 5.2, we only annotated the *events* and *timexes (temporal expressions)* for a document. For these entities, assume we have the relations shown

[7]Mapping between Allen's relations and TimeML relations are shown in Table 3.4.

```
The British government has formally <EVENT eid=
"e1">called</EVENT> for Sinn Fein, the IRA’s
political wing, to be <EVENT eid="e2">expelled
</EVENT> from the multiparty peace <EVENT
eid="e3"> talks</EVENT> on northern Ireland.
The <EVENT eid="e4">move</EVENT> had
been widely <EVENT eid="e5">expected</EVENT>
after northern Ireland police <EVENT eid="e6">
said</EVENT> they <EVENT eid="e7">believe
</EVENT> the IRA was be-hind two <EVENT
eid="e9">killings</EVENT> in Belfast <TIMEX3
 tid="t1">last week</TIMEX3>.
```

Figure 5.2: An excerpt from a TimeML annotated document

in Figure 5.3.

From the relations in Figure 5.3, we want to create a temporal structure so that we can easily make inferences about the implicit relations, i.e. identify how all entities are related to each other in terms of time.

With Timegraph, we create the temporal structure, which represents the entities to efficiently answer how all entities are related with each other temporally. The Timegraph representation with pseudo values and chain information

e1 AFTER e2	e4 AFTER e6
e1 AFTER e3	e6 SIMULTANEOUS e7
e1 SIMULTANEOUS e4	e6 AFTER t1
e4 SIMULTANEOUS e5	e9 IS-INCLUDED t1

Figure 5.3: Temporal relations (shown in easy to understand format) for the text in Figure 5.2

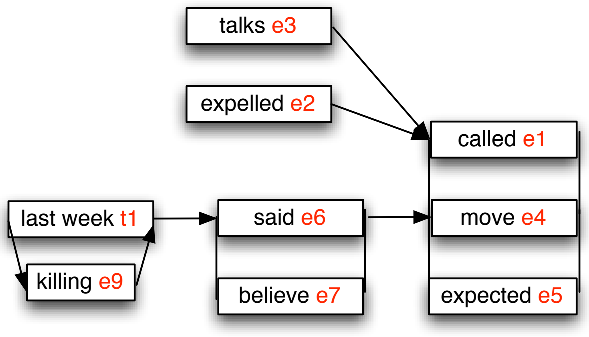

Figure 5.4: Intuitive temporal structure generated from the temporal relations in Figure 5.3

for temporal relations in Figure 5.3 is shown in Figure 5.5. As mentioned earlier, Timegraph only supports the point relations, hence intervals are converted to start and end points, e.g. e1 will be represented with *e1*s (start of e1) and *e1*e (end of e1).

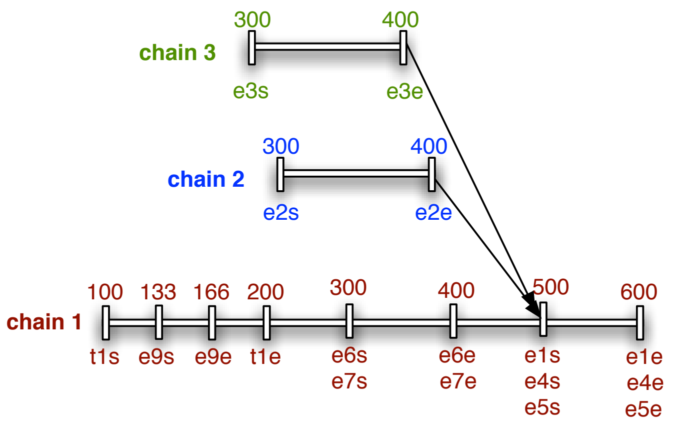

Figure 5.5: Timegraph representation of relations in Figure 5.3

With the pseudo value and chain information in the Timegraph, we can see in Figure 5.5 that it is easy to answer how two entities are related with each other.

5.3.3 Implementation

Timegraph is scalable. We ran the Timegraph construction algorithm on the complete TimeBank corpus and found that Timegraph construction time increases linearly with the increase of number of nodes and edges (= # of cross-chain links + # of chains) (Figure 5.6).

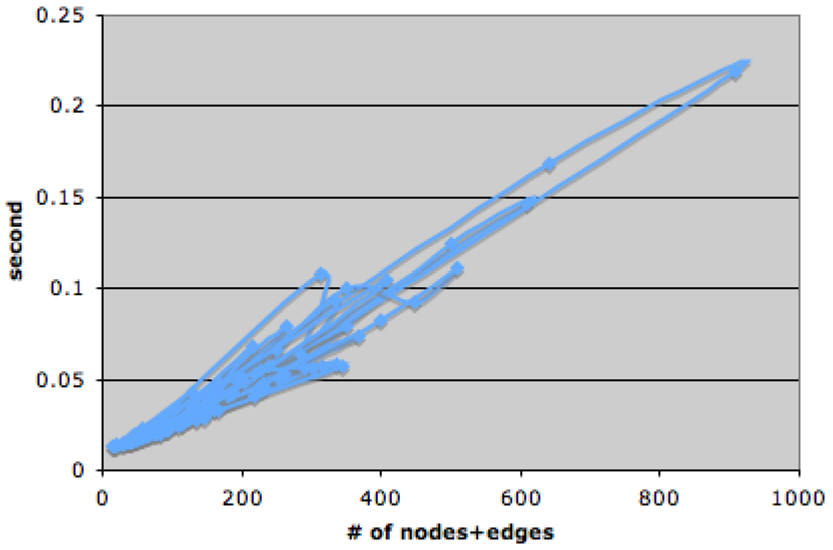

Figure 5.6: Number of nodes+edges (# of cross-chain links + # of chains) against time (in seconds) for Timegraph construction of all TimeBank documents.

The largest document, with 235 temporal relations (around 900 nodes+edges in Timegraph) only takes 0.22 seconds in a laptop computer with 4GB RAM and 2.26 GHz Core 2 Duo processor.

We also confirmed that the number of nodes + edges in Timegraph also increases linearly with number of temporal relations in TimeBank documents, i.e. our Timegraph construction time correlates with the # of relations in TimeBank documents (Figure 5.7).

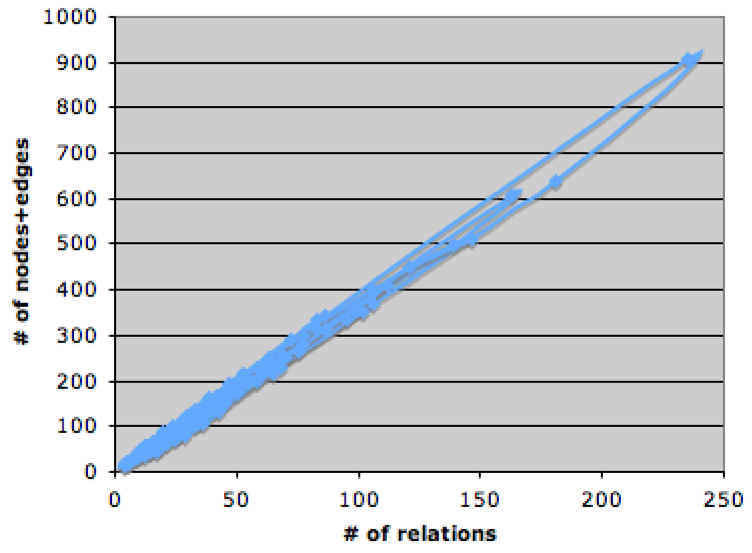

Figure 5.7: Number of temporal relations in all TimeBank documents against the number of nodes and edges in Timegraph of those documents.

Performance on searching in Timegraph is also shown in Figure 6.6), which also increases linearly against the number of relations and is computationally inexpensive.

In Timegraph, we cannot have any inconsistent relations, i.e. inconsistent relation will be ignored and will not be added in the Timegraph. As a result, if we have high confidence[8] relations, it is useful to add them first. Otherwise, Timegraph will discard valid high-confidence relations afterwards.

For example, we have C<A with confidence 0.4, A<B with confidence 0.8 and B<C with confidence 0.75. If we add relations in this particular order without considering the confidence, then we will ignore the relation B<C, since it is inconsistent with the existing relations C<A and A<B in the Timegraph

[8]We define the *confidence* score to represent the confidence for an automated system's claim. High confidence score means the automated system is confident about the claim and low confidence score means the automated system is suggesting the claim but it is not too confident about it.

(Figure 5.8).

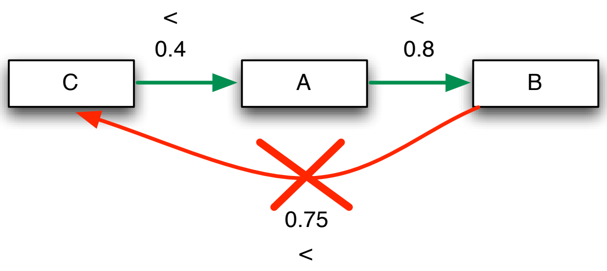

Figure 5.8: Example of discarding high-confidence relation in a Timegraph due of inconsistency

However, if we add the high-confidence relations first then we will discard C<A, which is a low-confidence relation (Figure 5.9).

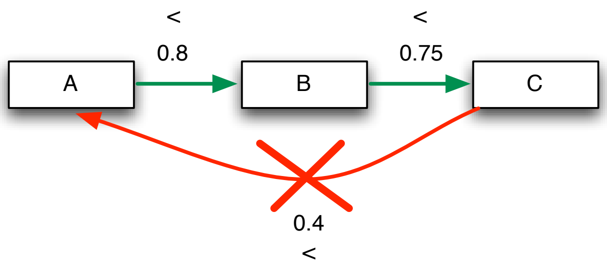

Figure 5.9: Example of adding high-confidence relations first in a Timegraph

Our Timegraph implementation can consider the confidence to add high-confidence relations first. However, our *temporal relation classification* module does not output the probability of different relations that we could feed in to the Timegraph construction module. However, in our proposed merging algorithms (section 7.2), we consider the weighted voting from different sources to calculate the confidence for each relation. In that case, we add the high confidence suggestions first and later add low confidence suggestions, as long as it is consistent with the Timegraph.

Another ideal case would be to classify the temporal relations considering the global inference (section 3.2.4). Capturing the closure properties in the classification task would ensure that all classified relations maintain the closure properties and they will not be discarded in the Timegraph.

5.4 Temporal Question-Answering with Temporal Reasoning

Our implemented temporal QA system is based on Timegraph (section 5.3.1). The Timegraph is created by adding the TimeML explicit relations. With the Timegraph's reasoning mechanism, the derived implicit relations are inferred. We can therefore answer both explicit and implicit temporal relations with Timegraph. To answer the questions about TimeML entities (based on time intervals) using Timegraph (based on time points), we convert the queries to point based queries.

For answering **yes/no questions**, we check the necessary point relations in Timegraph (relations in "Equivalent in Point Algebra" column of Table 5.1) to verify an interval relation. For example, to answer the question *is event1 after event2*, our system verifies whether *start(event1)* > *end(event2)*; if it is verified then the answer is *true*, if it conflicts with the Timegraph then it is *false*, otherwise it is *unknown*.

For answering **list and factoid questions**, the system traverses the Timegraph. For example, if we want to list all events before *event1*, our query to Timegraph would be to find all events that end before the start of *event1*.

With the pseudo value and chain information in the Timegraph it is easy to answer how two entities are related with each other. As a result, we can easily answer *yes/no, list* and *factoid questions*. We describe some examples below to demonstrate how we answer these questions.

For answering a *yes/no* question, *Was Fein called after the killings?* The Timegraph query will be `IS e1 AFTER e9`. We can see in the relations in Figure 5.3 that this particular relation is not explicitly mentioned, but with temporal reasoning we will be able to answer this implicit relation. In our Timegraph (Figure 5.5), both e1 and e9 are in the same chain 1. Hence we will just check if $e1\text{s} > e9\text{e}$, which is true, to answer this question.

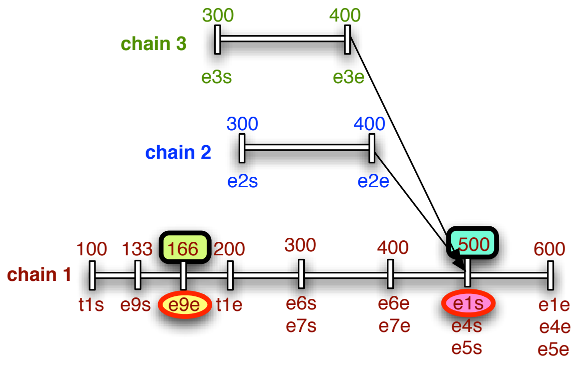

Figure 5.10: Showing $e1\text{s} > e9\text{e}$ in the Timegraph of Figure 5.5

We can also answer *list* questions by traversing the Timegraph. If we ask *What happened before calling Sinn Fein?* the Timegraph query will be `LIST BEFORE e1`. We will traverse through the Timegraph in all chains to answer $\{\forall\ ei\text{: } ei\text{e} < e1\text{s}\}$. In this case: e6, e7, t1, e9, e2 and e3.

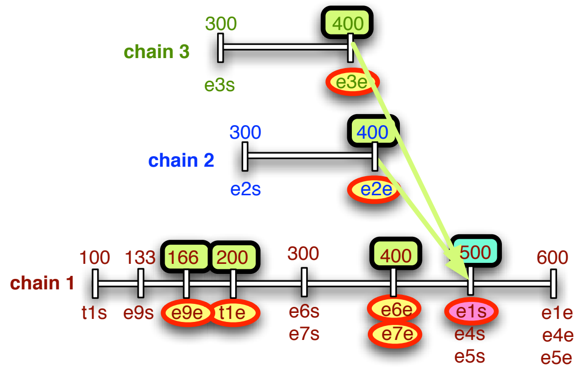

Figure 5.11: Showing {∀ ei: eie < $e1$s} in the Timegraph of Figure 5.5

For *factoid* question, we answer all entities that are simultaneous or are included in the entity. If we ask *When did the killing happened?* the Timegraph query will be `WHEN e9`. Our system will be able to answer during t1 (last week) by traversing the Timegraph. In this case, we will check entities (temporal expression and events) ei, such that, {∀ ei: eis ≤ $e9$s and eie ≥ $e9$e}.

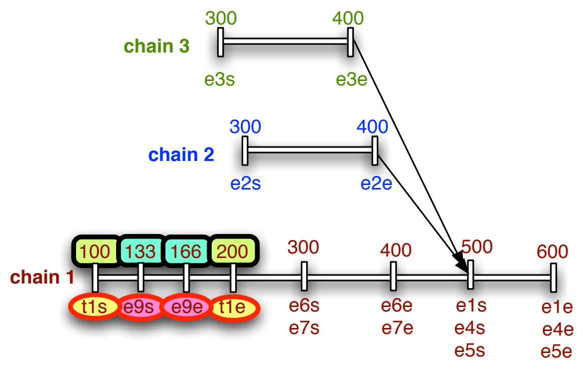

Figure 5.12: Showing {∀ ei: eis ≤ $e9$s and eie ≥ $e9$e} in the Timegraph of Figure 5.5

Currently, our system uses a specific syntax that represents human language questions. We do not convert natural language questions to our question syntax automatically, instead we input the questions in timegraph query syntax.

5.4.1 Evaluation

We created a set of 189 temporal questions (79 *yes/no*, 63 *list* and 47 *factoid questions*) from 25 TimeML annotated documents. These documents were randomly selected and the two volunteers came up with the questions by reading only the text, without looking at the existing temporal relations – TLINKS in the documents. As a result, it captures important time related questions from a document rather than all possible temporal relations. With these questions we wanted to evaluate how well our automated systems, TRIOS, can answer different question types. TRIOS is extracting TimeML annotations automatically from raw text (discussed in chapter 4) and with the help of our QA system (this chapter), based on Timegraph, we can answer these temporal questions. To evaluate TRIOS's capability to answer temporal questions from raw text, we take answers obtained by our QA system from the gold annotations as correct answers. Then, we compare these answers with those obtained from TRIOS. Table 5.2 reports the performance.

	Yes/No	List	Factoid
TRIOS	34.18	37.03	22.04

Table 5.2: Performance in TRIOS on different temporal question types

Next, we evaluate how many yes/no questions can be answered with explicit TimeML relations, i.e. without temporal reasoning. To determine this, we check if we can answer the question directly from temporal relations of annotation (explicit) or whether we have to use timegraph (implicit) to make temporal inference. Out of our 79 yes/no questions, the gold annotation was capable of answering only 42 questions in the first place. Out of these 42 questions, we found only 7 (16.67%) were explicitly annotated and rest (83.3%) were answered with temporal inference. The rest were *unknown*, i.e. the human annotation did not provide the information needed to make the inference. This performance is reported in Table 5.3.

Gold annotation answered from explicit annotation	7/42
Gold annotation answered with implicit annotation (making temporal inference with timegraph)	35/42
Gold annotation unable to support needed inference	37/79

Table 5.3: Statistics of explicit vs implicit questions

This statistics in Table 5.3 suggest that gold annotation is not complete enough to answer many questions and we need to do temporal reasoning to answer the majority of the questions.

More detailed evaluation and thorough analysis of our QA system on TimeML annotated documents can be found in section 6.2, where we use our QA system to evaluate systems extracting temporal information.

5.5 Summary

We present a **question-answering** (QA) system capable of temporal reasoning, given a TimeML annotated document. Our QA system creates the **temporal structure** using a Timegraph (MS90), which helps our QA system to infer the implicit temporal relations from the explicit ones. With the help of the temporal structure, our QA system answers *yes/no*, *list* and *factoid* questions. Since our system is generic, it can be used to answer temporal questions about any document annotated in TimeML.

Our experiments indicate that the vast majority of the temporal questions cannot be answered by the available explicit relations between entities annotated in the corpora. Answering these questions requires temporal reasoning to obtain the implicit relations between entities.

6 The Evaluation of Temporal Information

In this chapter, we propose new metrics to evaluate temporal annotation and temporal information understanding.

Prior evaluation methods (section 3.3) for different temporal information extraction (TIE) subtasks have borrowed *precision* and *recall* measures from the Information Retrieval community, which are not completely suitable for evaluating temporal annotations. To resolve this issue, we propose a new method to evaluate temporal annotations, which has been adopted to evaluate the TempEval 2013 participants. Our new metric considers semantically similar, but distinct, temporal relations and consequently gives a single score that could be used to identify the overall temporal awareness of a system.

However, evaluating temporal annotations is not the best way to evaluate a system's capabilities of understanding temporal information. Thus, we propose an additional approach, using temporal question-answering, to better evaluate the temporal information understanding.

6.1 Evaluation of Temporal Annotation

We propose a new method for evaluating systems that extract temporal information from text. We use temporal closure to reward relations that are equivalent but distinct. Our metric measures the overall performance of systems with a single score, making comparison between different systems straightforward. Our approach is easy to implement, intuitive, scalable and computationally inexpensive. Our proposed metric has been adopted for evaluating participants at TempEval-3.

6.1.1 Motivation

Prior evaluation methods (section 3.3) for different TIE subtasks have borrowed precision and recall measures from the information retrieval community. This has two problems: First, systems express temporal relations in a different, yet equivalent, ways. Consider a scenario where the reference annotation contains e1<e2 and e2<e3 and the system identifies the relation e1<e3. The traditional evaluation metric will fail to identify e1<e3 as a correct relation, even though it is a logical consequence of the reference annotation. Second, traditional evaluations tell us how well a system performs in a particular task, but not the overall performance. For example, in TempEval-2 there were 6 subtasks (*event extraction, temporal expression extraction* and 4 subtasks on *identifying temporal relations*). If different systems perform best in different subtasks, we cannot compare overall performance of systems.

Using Timegraph, we use temporal closure to identify equivalent temporal relations and produce a single score that measures the temporal awareness of each system.

6.1.2 Related Work

To calculate the inter-annotator agreement between annotators in the temporal annotation task, some researchers have used semantic matching to reward distinct but equivalent temporal relations. Such techniques can equally well be applied to system evaluation.

Setzer et al. (2003) (SGH03) use temporal closure to reward equivalent but distinct relations. Consider the example in Figure 6.1 (due to Tannier and Muller, 2008 (TM08)). Consider graph K as the reference annotation graph, and S1, S2 and S3 as outputs of different systems. The bold edges are the extracted relations and the dotted edges are derived. The traditional matching approach will fail to verify that B<D is a correct relation in S2, since there is no explicit edge between B and D in the reference annotation (K). But a metric using temporal closure would create all implicit edges and be able to reward B<D edge in S2.

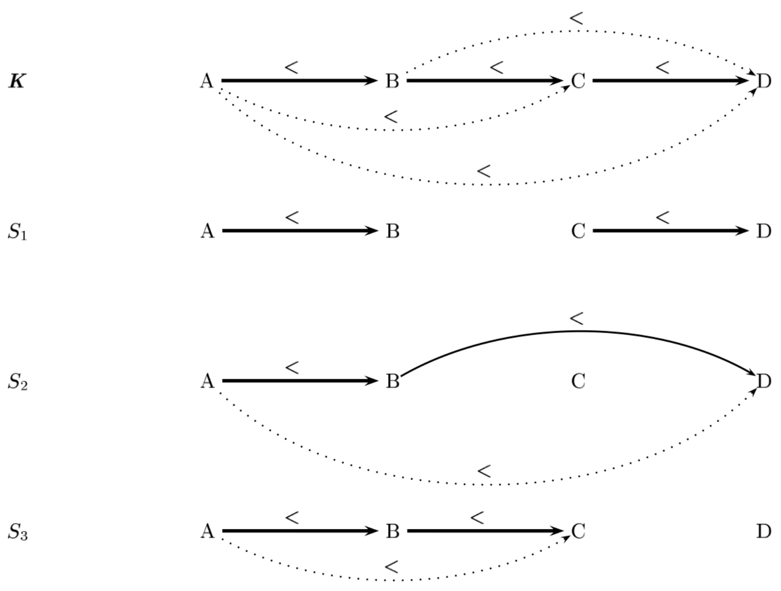

Figure 6.1: Examples of temporal graphs and relations

Setzer et al.'s approach works for this particular case, but as pointed by

Tannier and Muller (2008), it gives the same importance to all relations, whereas some relations are not as crucial as others. For example, with K again as the reference annotation, S2 and S3 both identify two correct relations, so both should have a 100% precision, but in terms of recall, S3 identified 2 explicit relations and S2 identified one explicit and one implicit relation. With Setzer at al.'s technique, both S2 and S3 will get the same score, which is not accurate.

Tannier and Muller handle this problem by finding the core[1] relations. For recall, they consider the reference core relations found in the system core relations and for precision they consider the system core relations found in the reference core relations. They noted that core relations do not contain all information provided by closed graphs. Hence their measure is only an approximation of what should be assessed. Consider the previous example again. If we are evaluating graph S2, they will fail to verify that B<D is a correct edge.

We have shown that both of these existing evaluation metrics reward relations based on semantic matching, but still fail in specific cases.

6.1.3 Evaluation of Temporal Annotation

We also use temporal closure to reward equivalent but distinct relations. However, we do not compare against the temporal closure of reference annotation and system output, like Setzer et al., but we use the temporal closure to verify if a temporal relation can be derived or not. Our precision and recall is defined as:

[1]For relation $R_{A,B}$ between A and B, derivations are $R_{A,C}$, $R_{B,C}$, $R_{A,D}$, $R_{B,D}$. If the intersection of all these derived relations equals $R_{A,B}$, it means that $R_{A,B}$ is not a core relation, since it can be obtained by composing some other relations. Otherwise, the relation is a core, since removing it tends to loss of information.

$$Precision = \frac{|Sys_{relation} \cap Ref^{+}_{relation}|}{|Sys_{relation}|} \tag{6.1}$$

$$Recall = \frac{|Ref_{relation} \cap Sys^{+}_{relation}|}{|Ref_{relation}|} \tag{6.2}$$

where, G^{+} is the closure of graph G.

We calculate the $Precision$ by checking the number of system temporal relations ($Sys_{relation}$) that can be verified from the reference annotation temporal closure graph ($Ref^{+}_{relation}$), out of number of temporal relations in the system output ($Sys_{relation}$). Similarly, we calculate the $Recall$ by checking the number of reference annotation temporal relations ($Ref_{relation}$) that can be verified from the system output's temporal closure graph ($Sys^{+}_{relation}$), out of number of temporal relations in the reference annotation ($Ref_{relation}$).

The harmonic mean of precision and recall, i.e. fscore, will give an evaluation of the temporal awareness of the system.

As an example, consider again the examples in Figure 6.1, with K as reference annotation. S1 and S3 clearly have 100% precision, and S2 also gets 100% precision, since the B<D edge can be verified through the temporal closure graph of K. Note, our recall measure does not reward the B<D edge of S2, but it is counted for precision. S1 and S3 both get a recall of 2/3, since 2 edges can be verified in the reference temporal closure graph. This scheme is similar to the MUC-6 scoring for co-reference resolution (Vilain et al., 1995) (VBA^{+}95). Their scoring estimated the minimal number of missing links necessary to complete a co-reference chain in order to make it match the human annotation. Here in both S1 and S3, we are missing one edge to match with the reference annotation; hence 2/3 is the appropriate score. Precision, recall and fscore for all these system output are shown in Table 6.1.

System	Precision	Recall	Fscore
S1	2/2=1	2/3=0.66	0.8
S2	2/2=1	1/3=0.33	0.5
S3	2/2=1	2/3=0.66	0.8

Table 6.1: Precision, recall and fscore for systems in Figure 6.1 according to our evaluation metric

6.1.4 Implementation

Our proposed approach is easy to implement with an existing temporal closure implementation. We used our Timegraph implementation (section 5.3.1) to verify relations in the closure graph.

6.1.5 Evaluation

Our proposed evaluation metric has some very good properties, which makes it very suitable as a standard metric. This section presents a few empirical tests to show the usefulness of our metric.

Our precision and recall goes with the same spirit with traditional precision and recall, as a result, performance decreases with the decrease of information. Specifically,

1. If we remove relations from the reference annotation and then compare that against the full reference annotation, then recall decreases linearly. Shown in Figure 6.2.

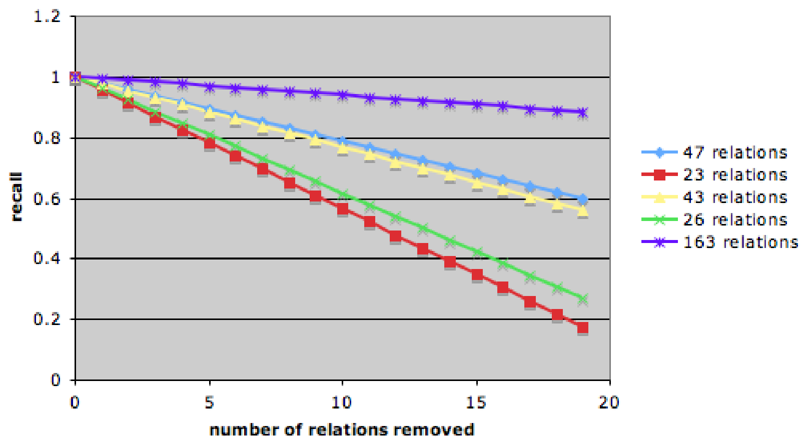

Figure 6.2: For 5 TimeBank documents, the graph shows performance drops linearly in recall by removing temporal relations one by one.

2. If we introduce noise by adding new relations, then precision decreases linearly (Figure 6.3).

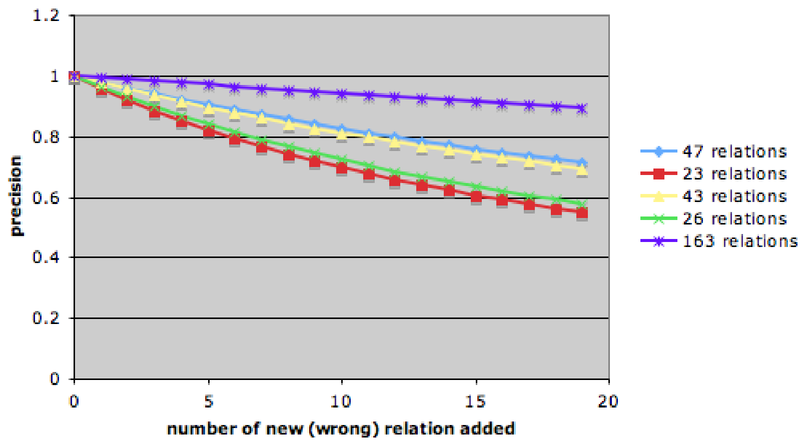

Figure 6.3: For 5 TimeBank documents, the graph shows performance drops linearly in precision by adding new (wrong) temporal relations one by one.

3. If we introduce noise by changing existing relations then fscore decreases linearly (Figure 6.4).

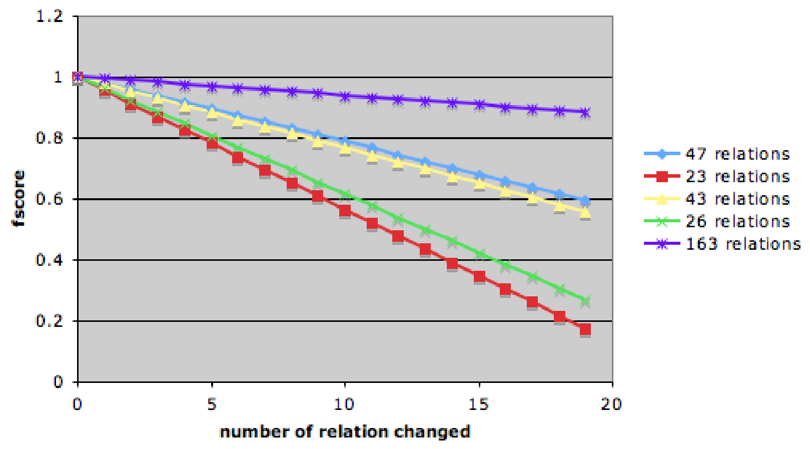

Figure 6.4: For 5 TimeBank documents, the graph shows performance drops linearly in fscore by changing temporal relations one by one.

4. If we remove temporal entities (such as events or temporal expressions), performance decreases more for entities that are temporally related to more entities. This means, if the system fails to extract important temporal entities then the performance will decrease more (Figure 6.5).

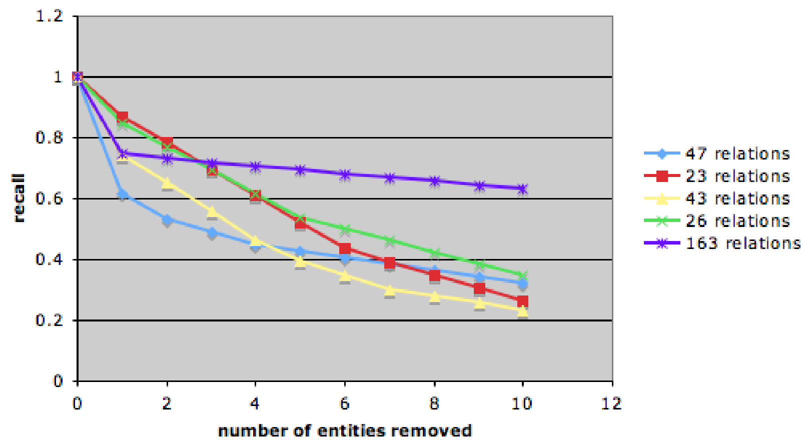

Figure 6.5: For 5 TimeBank documents, performance drop in recall by removing temporal entities.

In our experiment, we remove the temporal entities related with a maximum number of entities first. It is evident from the graph that performance decreased more for removing important entities (first few entities).

These properties explain that our final fscore captures how well a system extracts *events, temporal expressions* and *temporal relations*. Therefore this single score captures all the scores of six subtasks in TempEval-2, making it very convenient and straightforward to compare different systems.

Our Timegraph implementation is also scalable (shown in section 5.3.3). Searching in Timegraph, which we need for temporal evaluation, also depends on number of nodes and edges, hence number of TimeBank relations. We ran a temporal evaluation on TimeBank corpus using the same document as system output. The operation included creating two Timegraphs and searching in the Timegraph. As expected, the searching time also increases linearly against the number of relations and is computationally inexpensive (Figure 6.6).

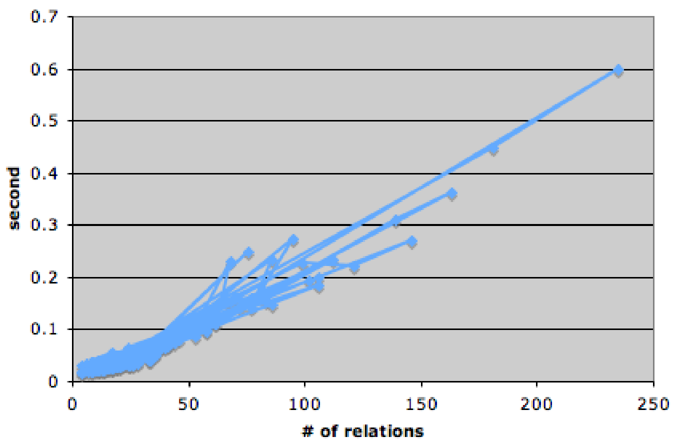

Figure 6.6: Number of relation against time (in seconds) for all documents of TimeBank corpus.

These properties explain why our metric is very suitable as a standard metric. In the next section, we explain where our metric falls short and additionally we also propose an updated solution.

6.1.6 Problem with our Existing Solution

Our proposed solution works perfectly to identify the number of explicit relations (links) missing to complete the human annotation by semantic matching. However, it fails[2] to distinguish if some systems are extracting more implicit relations than other systems. These implicit relations are extra information that other systems are missing and can be verified from the temporal closure of the reference annotation.

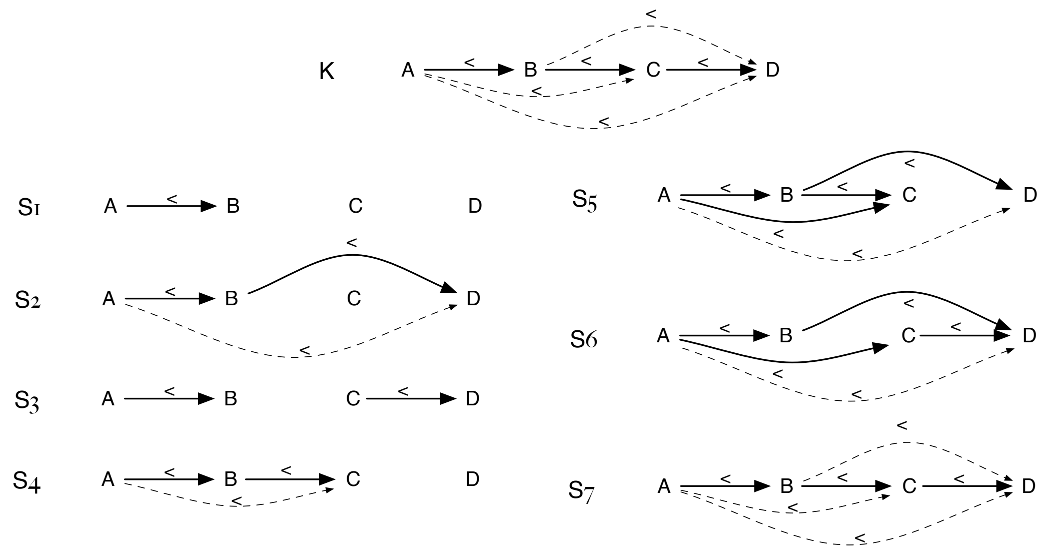

Figure 6.7: Reference annotation K and system annotations S1-S7

Consider the examples in Figure 6.7 to clarify the problem, where K is the reference annotation. Explicit relations are shown as solid lines and the inferred relations as dashed lines. Both S1 and S2 get 100% in precision since their extracted relations can be verified in the closure of reference annotation.

[2]This problem was identified by Professor Lenhart K. Schubert

Both of them also extract one explicit relation (A<B) and miss two explicit relations, i.e. if we include two more relations then we can get the same annotation as the reference annotation. Even though S2 has one extra relation, to get to the reference annotation, it still needs to add two relations like S1. Hence, they get same score according to our proposed solution in section 6.1.3. However, S2 is extracting some extra implicit information, which makes it better than S1, even though still worse than S4 (since we can get the reference annotation from S4 just by adding one relation). In this section, we propose an extension to our proposed solution in section 6.1.3 to make these distinctions. At the same time, we want our proposed solution in section 6.1.3 to reward S3 and S4 equally, even though S4 has one extra relation. Here this extra relation is found by taking the closure of existing relations; it is not giving any extra information, such as S2's B<D relation against S1. And in both cases, we need another relation to get the reference annotation; hence, both S3 and S4 would be rewarded equally. We also make sure this property holds in our updated solution.

6.1.7 Proposed Updated Solution

Following are the issues we want to consider in our updated solution:

1. Maintain the properties of our proposed solution in section 6.1.3 to reward explicit relations appropriately.

2. Not to reward implicit relations that can be derived from the existing core relations. To handle this we consider the reduced graph, which is derived from the original graph by removing all temporal relations that do not cause a loss in temporal information, i.e. we keep all core relations and remove only the relations that can be derived from the core relations. For example, assume we have A<B, B<D, A<D relations in

a graph (consider S2 in Figure 6.8). In our reduced graph we will only have A<B and B<D. A<D will be removed from the reduced graph since we can derive it with the temporal closure.

The Timegraph (MS90) algorithm maintains the reduced graph and infers implicit relations. Our proposed solution in section 6.1.3 depends on Timegraph for temporal closure, hence we take benefit of Timegraph for reduced graph implementation too (check section 5.3.1 for details).

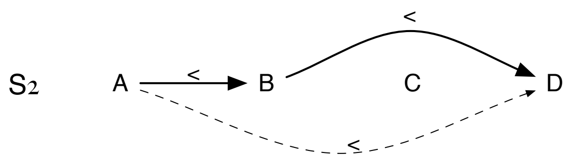

Figure 6.8: Graph S2 showing extra implicit relation B<D

3. Given the reduced graph and the closure graph, we can easily find the extra implicit relations that we want to reward.

   Given our reference graph K, for graph S2, A<B is a core relation. B<D is an implicit relation in graph K, but it is a core relation for S2, which we want to reward. A<D is an implicit relation for S2 that we can derive from core relations, so we do not want to reward it. We would loosely refer A<B relation as explicit relation and B<D as implicit relation in the rest of the section.

4. We want to reward B<D to distinguish with other systems (e.g. against S1 in Figure 6.7), but we do not want to reward B<D equally as A<B or core/explicit relations of the reference graph.

Considering all these issues, we developed the following formula.

$$Precision = \frac{|Sys^{-}_{relation} \cap Ref^{+}_{relation}|}{|Sys^{-}_{relation}|} \tag{6.3}$$

$$Recall = \frac{|Ref^-_{relation} \cap Sys^+_{relation}| + w * |(Sys^-_{relation} - Ref^-_{relation}) \cap Ref^+_{relation}|}{|Ref^-_{relation}|} \tag{6.4}$$

Where, $w = \frac{0.99}{(1+|\#Ref^+_{relation}|-|Ref^-_{relation} \cap Sys^+_{relation}|)}$, described below.

Our proposed updated solution has the following criteria:

1. G^+ is the closure graph of G and G^- is the reduced graph of G. Our precision formula and first part of recall formula remains same as our proposed solution in section 6.1.3. The only difference is, here we are considering the reduced graph, whereas the earlier solution was considering the actual graph. Reduced graph is more appropriate, since it does not reward or penalize for redundant information. This part of the formula maintains the properties of that solution.

2. For recall, we calculate explicit relations and implicit relations separately. The recall score will sort the systems in terms of the explicit relations first and then based on the implicit relations. That is, our improvement will distinguish systems that extract more implicit relations with equal number of explicit relations. For example, in Figure 6.7, S2 will get higher recall than S1 and S4 will get higher recall than S2. For clarification, $(Ref^-_{relation} \cap Sys^+_{relation})$ captures system explicit relations that are found in gold explicit relations; $(Sys^-_{relation} - Ref^-_{relation})$ captures implicit system relations in terms of $Ref^-_{relation}$, this will capture B<D in S2; $(Sys^-_{relation} - Ref^-_{relation}) \cap Ref^+_{relation}$ verifies the extra implicit relation is a valid implicit relation in terms of reference closure graph.

3. Finally, we formulate w in a way so that we always sort with the explicit relation count first. We define, $w = \frac{0.99}{(1+|\#Ref^+_{relation}|-|Ref^-_{relation} \cap Sys^+_{relation}|)}$, where the denominator captures the maximum number of implicit relations possible.

$(Ref^{-}_{relation} \cap Sys^{+}_{relation})$ captures the number of explicit relations for Sys. In Figure 6.7, for S2, it will contain only A<B.

$\#Ref^{+}_{relation}$ captures maximum possible relations for entities in $Sys^{-}_{relation}$. $Ref^{+}_{relation}$ could be approximated with $\frac{1}{2} * n * (n-1)$, maximum possible relations with n entities, where $n(>=0)$ is the number of temporal entities, events or temporal expressions in $Ref^{-}_{relation}$. For $\#Ref^{+}_{relation}$, we consider $n = |Sys^{-}_{entity} \cap Ref^{-}_{entity}|$, this way we do not consider extra entities that do not contribute to the maximum relations possible for $Sys^{-}_{relation}$. $|(Sys^{-}_{relation} - Ref^{-}_{relation}) \cap Ref^{+}_{relation}|$ is the implicit relations extracted by the system. We capture the maximum possible implicit relations (total relations - core relations) with the denominator of w. Hence, $w * |(Sys^{-}_{relation} - Ref^{-}_{relation}) \cap Ref^{+}_{relation}|$ is at most 0.99. For $n \leq 2$, the number of maximum possible relations can be equal to maximum core relations, i.e. it might not have any implicit relations. In that case, $(|\#Ref^{+}_{relation}| - |Ref^{-}_{relation} \cap Sys^{-}_{relation}|)$ equals to 0, hence the additional +1 in the equation.

4. If all explicit relations exist in $Sys^{-}_{relation}$ graph then $Sys^{-}_{relation} = Ref^{-}_{relation}$, hence $Sys^{-}_{relation} - Ref^{-}_{relation} = \{\}$ and Recall = 1.

5. $Sys^{-}_{relation} - Ref^{-}_{relation}$ will be non-empty if there are implicit relations. It happens if we are missing any explicit relations. If at least one explicit relation is missing, $w * |(Sys^{-}_{relation} - Ref^{-}_{relation}) \cap Ref^{+}_{relation}$ is at most 0.99.

6.1.8 Evaluation of Updated Solution

Given that our improvement could use the existing temporal closure to verify the implicit relations, our updated evaluation metric also maintains the criteria of our proposed solution in section 6.1.3, which are:

1. Our measure decreases with the decreasing of information (in a monotonic and regular way, linearly with the level of information or correctness provided).

2. Evaluation time for a document linearly increases with the number of relations in the document.

To understand the difference when accounting for implicit relations, we consider our formula, described in section 6.1.3, with reduced graph to match with our new proposed metric.

$$Precision = \frac{|Sys^{-}_{relation} \cap Ref^{+}_{relation}|}{|Sys^{-}_{relation}|} \tag{6.5}$$

$$Recall = \frac{|Ref^{-}_{relation} \cap Sys^{+}_{relation}|}{|Ref^{-}_{relation}|} \tag{6.6}$$

To evaluate the performance of our updated solution and to compare against other evaluation metrics, we artificially created some graphs. In the reference graph (annotation), all the nodes are connected in a way, such that $\{\forall i | N_i < N_{i+1}\}$, where N_i is the i^{th} node. In graph G1, we have few core relations. In G2, we added some implicit relations, which are useless in the sense that it does not add any extra information, i.e. it can be derived from the existing core relations. Finally, in G3, we added some useful implicit relations, which are new information, i.e. cannot be derived from the existing core relations. All these annotations (graphs) are shown in Figure 6.9.

We compare the performance of system annotations (G1, G2, G3) using five evaluation metrics (Table 6.2) and show the performance in Table 6.3 and 6.4.

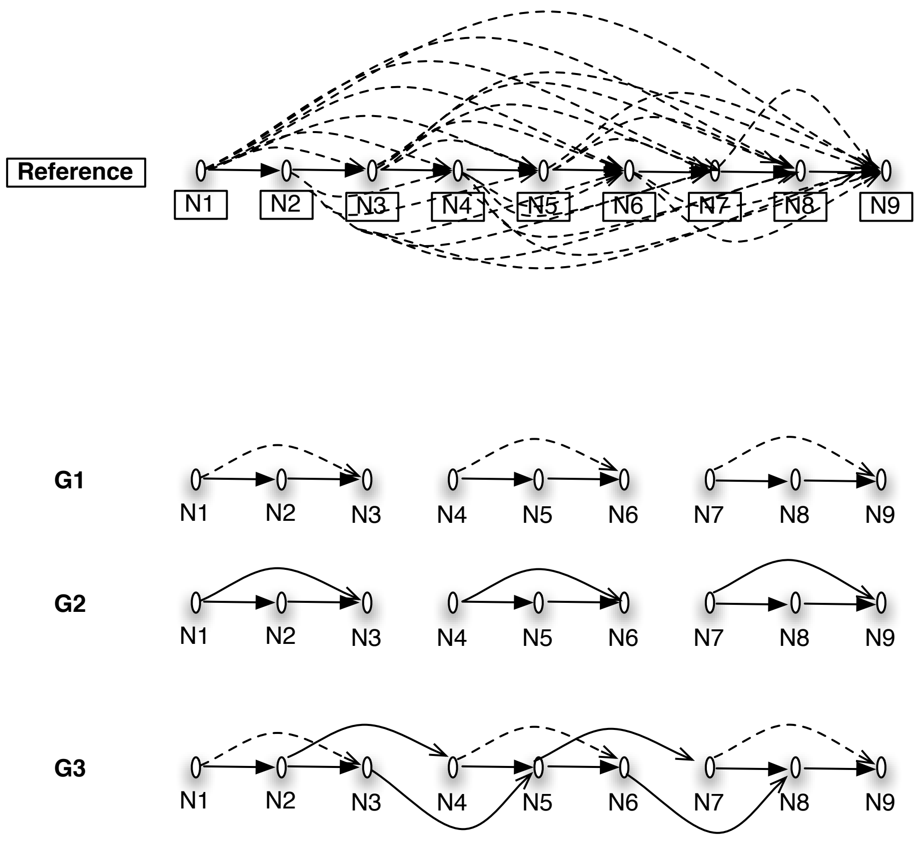

Figure 6.9: Reference annotation and System annotations (G1, G2, G3) (with number of nodes, n = 9) to compare temporal evaluation metrics

Evaluation Metric	Recall	Precision
TempEval-2 (VSCP10)	$\frac{Ref \cap Sys}{Ref}$	$\frac{Sys \cap Ref}{Sys}$
Setzer et al. (SGH03)	$\frac{Ref^{+} \cap Sys^{+}}{Ref^{+}}$	$\frac{Sys^{+} \cap Ref^{+}}{Sys^{+}}$
Tannier and Muller (TM08)	$\frac{Ref^{-} \cap Sys^{-}}{Ref^{-}}$	$\frac{Sys^{-} \cap Ref^{-}}{Sys^{-}}$
Our ACL'11 metric (Eqn 6.5)	$\frac{\lvert Ref^{-} \cap Sys^{+} \rvert}{\lvert Ref^{-} \rvert}$	$\frac{\lvert Sys^{-} \cap Ref^{+} \rvert}{\lvert Sys^{-} \rvert}$
Our Updated metric (Eqn 6.3)	$\frac{\lvert Ref^{-} \cap Sys^{+} \rvert + w * \lvert (Sys^{-} - Ref^{-}) \cap Ref^{+} \rvert}{\lvert Ref^{-} \rvert}$ [1]	$\frac{\lvert Sys^{-} \cap Ref^{+} \rvert}{\lvert Sys^{-} \rvert}$

[1] where, $w = \frac{0.99}{(1 + \lvert \# Ref^{+} \rvert - \lvert Ref^{-} \cap Sys^{+} \rvert)}$

Table 6.2: Metrics for Temporal Evaluation

	few core relations - G1		*adding useless implicit relations - G2*		*adding **useful** implicit relations - G3*	
	G1 Recall	G1 Precision	G2 Recall	G2 Precision	G3 Recall	G3 Precision
TempEval-2 (VSCP10)	0.75	1	0.75	0.66	0.75	0.6
Setzer et al. (SGH03)	0.25	1	0.25	1	0.3611	1
Tannier and Muller (TM08)	0.75	1	0.75	1	0.75	0.6
ACL'11 metric (Eqn 6.5)	0.75	1	0.75	1	0.75	1
Updated metric (Eqn 6.3)	0.75	1	0.75	1	**0.7659**	1

Table 6.3: Performance Comparison for graphs in Figure 6.9 with n=9 nodes

	few core relations - G1		*adding <u>useless</u> implicit relations - G2*		*adding **useful** implicit relations - G3*	
	G1 Recall	G1 Precision	G2 Recall	G2 Precision	G3 Recall	G3 Precision
TempEval-2 (VSCP10)	0.6896	1	0.6896	0.66	0.6896	0.5263
Setzer et al. (SGH03)	0.0689	1	0.0689	1	0.1103	1
Tannier and Muller (TM08)	0.6896	1	0.6896	1	0.6896	0.5263
ACL'11 metric (Eqn 6.5)	0.6896	1	0.6896	1	0.6896	1
Updated metric (Eqn 6.3)	0.6896	1	0.6896	1	**0.6911**	1

Table 6.4: Performance Comparison for graphs in Figure 6.9 with n=30 nodes

From the Table 6.3 and 6.4 we observe the following:

1. TempEval-2 evaluation metric (VSCP10) fails when we have any implicit relations, since the matching is exact matching instead of semantic matching.

2. Tannier and Muller (2008) (TM08)) showed Setzer et al. (2003) (SGH03) gives equal weights to all relations (both implicit and explicit), hence it gives a misleading lower score for evaluation. It is even more evident when we have more relations (check recall scores for Setzer et al. in Figure 6.4).

3. Except TempEval-2 metric, all other metrics' score remained same when we added <u>useless</u> implicit relations (difference between G1 and G2), since all other metrics are doing semantic matching.

4. When **useful** implicit relations are added, we would expect the evaluation metric to reward systems. We find that performance using our ACL'11 metric remained same, and it even decreased in precision for TempEval-2 metric and Tannier and Muller's metric. However, it increased for both Setzer et al.'s solution and our improved solution.

Comparing the performance of different systems and our observations, it is evident that Setzer et al. and our updated metric gives higher performance for better systems by rewarding the implicit relations. However, Setzer et al.'s score is practically unusable due to its score range (check recall scores for Setzer et al. in Figure 6.4). On the other hand, our updated metric's improvement for adding **useful** implicit relations is not very significant either, but it is comparable to our earlier metric (equation 6.5), which has been adopted to evaluate TempEval participants.

Next, we report the performance of two state-of-the-art systems, TIPSem (LSNC10) and TRIOS (chapter 4) in TimeBank 1.2 corpus (PHS$^+$03) using both our ACL'11 metric (Table 6.5) and our updated metric (Table 6.6) for comparison.

	Precision (%)	Recall (%)	Fscore (%)
TIPSem	27.2048	35.8517	30.9354
TRIOS	26.6020	27.2986	26.9458

Table 6.5: Performance of TIPSem and TRIOS with equation 6.5, metric similar to our proposed solution in section 6.1.3.

	Precision (%)	Recall (%)	Fscore (%)
TIPSem	27.2048	35.8771	30.9448
TRIOS	26.6020	27.3528	26.9722

Table 6.6: Performance of TIPSem and TRIOS with our updated metric (Equation 6.3 and 6.4).

Comparing Table 6.5 and Table 6.6, we can see that, in terms of numbers, the difference is insignificant (+0.2% improvement in recall) from our proposed solution in section 6.1.3. However, this difference is important, since (i) we only distinguish systems with same number of explicit relations but different number of implicit relations, and (ii) we are considering the maximum reward for all implicit relations in a document less than the reward of just one explicit relation.

6.1.9 Comments on the Updated Solution

Even though our updated solution is based on the idea that reward for all implicit relations will be less than one explicit relation, we think this idea is partially flawed.

For instance, we take a set of explicit relations and generate a new graph where we generalize the explicit relations slightly – e.g. we change every *before* relation to (*before* OR *meet*)[3]. That gives us a new graph with much useful information in it - but it will get a worse score than a minimal graph that has just one of those explicit relations changed back to its original value and deleting all the others.

Future research in this area should explore this problem when proposing a updated metrics.

6.2 Evaluating Temporal Understanding with Question Answering

In the previous section we proposed evaluation metrics for evaluating temporal annotations. In this section we propose a metric to evaluate the temporal

[3]Note this disjunction can be captured in a point-based logic. In point-algebra X *before* Y is $X_{start} < X_{end} < Y_{start} < Y_{end}$ and X *meet* Y is $X_{start} < X_{end} = Y_{start} < Y_{end}$. Hence, in point-algebra, we can easily capture X *before* OR *meet* Y with $X_{start} < X_{end} \leq Y_{start} < Y_{end}$. However, in our Timegraph implementation, we do not capture the disjunctions.

information understanding.

6.2.1 Motivation

The initial purpose of the temporal annotation scheme, TimeML (PCI$^+$03), was to handle the complex temporal question answering (QA) task. Given the complexity of temporal QA, the focus was shifted to solve the automated TimeML annotation subtasks. Currently, the automated systems solving these subtasks are evaluated using corpus-based approach (section 3.3 and 6.1). On the other hand, task based evaluations have not been proposed for temporal information processing.

Corpora annotated in TimeML are certainly needed for developing and training automated systems. However, we argue that measuring how well automated systems understand the temporal aspects of language with a corpus-based evaluation is not the most appropriate evaluation. Temporal question answering makes a much better evaluation. Our main arguments are as follows:

1. Answering questions is a natural way of evaluating language understanding for humans. Using temporal question answering, we can better judge how well systems understand the temporal information of a document.

2. Creating temporal questions is much easier and less time-consuming for humans than annotating temporal information. It makes it possible to easily create large test-sets and also to evaluate the generality of systems in new domains.

3. Corpus-based performance may not reflect how well systems can understand important temporal information. Since humans ask questions about relevant information, QA scores better capture the understanding

of important temporal information as compared to corpus-based evaluation where all information is equally important for scoring.

4. Human annotations can be incomplete. Evaluating automated systems against incomplete annotation might not reflect the actual performance of automated systems. With QA, we can also evaluate: (i) how complete the available human annotations are and (ii) how automated systems are performing compared to human annotations.

We presented our temporal QA system (in Section 5.4) that handles temporal reasoning. Our system allows answering complex temporal questions about a text annotated with TimeML. In this section we will show that our system can be used for evaluation of temporal understanding.

6.2.2 Temporal QA as Evaluation

We created a set of 189 temporal questions (79 *yes/no*, 63 *list* and 47 *factoid questions*) from 25 TimeML annotated documents. With these questions we evaluated how accurate the annotations of automated systems are against gold human annotation, i.e. how these systems understand these documents in terms of temporal information. To evaluate that, we take answers obtained by our QA system from the gold annotations as correct answers. Then, we compare these answers with those obtained from automated annotations. As automated systems, we consider TIPSem (LSNC10) and TRIOS (chapter 4) from TempEval-2 (VSCP10). Table 6.7 reports the QA results in addition to the results of these systems using our proposed metric in section 6.1.3, which gives a single score for corpus-based evaluation (corpus-score).

	Yes/No	List	Factoid	Corpus-score
TIPSem	48.10	43.49	53.30	31.60
TRIOS	34.18	37.03	22.04	27.16

Table 6.7: Performance in Temporal QA and corpus based score against gold annotation

In our experimental data, we find that TIPSem does better than TRIOS in every category of questions which matches corpus-based score. However, we can see that TIPSem does significantly better than TRIOS in factoid question. Corpus-scores cannot distinguish on how well systems actually perform real tasks like answering factoid questions. By evaluating with temporal QA we can understand the detailed capabilities of systems.

In the next experiment we manually answered the yes/no questions and compared the performance of systems and gold annotation against human answers to understand the coverage of human temporal annotations and automated temporal annotations. The results are reported in Table 6.8.

	Gold	TIPSem	TRIOS
Comparing Yes/No answers against human answers	48.10	37.97	22.78
Comparing No/Yes answers against TimeML gold annotations	100	48.10	34.17

Table 6.8: Comparison of gold and system annotations against human answers

We found that performance for gold annotation is not 100% (it is only 48%). This is because the gold annotation does not have complete temporal information coverage, i.e. it does not have all necessary relations to allow the reasoning required to answer the questions. It is important to note that with our QA system, an annotation with complete coverage will have a 100% performance for these questions.

Another thing to note is, when comparing against human answers, the difference between the gold performance and system performance is much lower. This result is suggestive that automated systems can perform very close to human annotations currently available.

We also notice that all scores decrease from the performance reported in Table 6.7. This is because many *yes/no questions* cannot be answered from the gold annotation, i.e. the gold annotation did not have all necessary relations to make the inference. When the systems also answered *unknown* in those cases, they had a match. Hence, when compared against the exact human answers (usually not *unknown*), all the scores decreased.

Finally, we wanted to find out if there are some questions that one of the systems could answer but was not answerable from the gold annotations. We found that there were 11 such instances for TIPSem system and 12 such instances for TRIOS system. For example, one such instance was the *yes/no* example we showed in the section 5.4 - `IS e1 AFTER e9`. The gold annotation had the relation `e1 SIMULTANEOUS e4`, but it was missing other necessary relations to infer the relationship between e1 and e9. However, one system was able to answer this particular question. This experiment also strengthens our claim about incompleteness of human temporal annotations.

The incompleteness of human temporal annotation is also discouraging for evaluating automated systems in corpus-based evaluation. Since we will end up penalizing automated systems for some good extraction as well. All human annotation suffers this problem, but in this case, annotating temporal information is much harder than answering some temporal questions from text, i.e. less chance to have wrong human answers than wrong human annotation. As a result, this is an additional advantage of evaluating temporal understanding with QA compared to corpus-based annotation evaluation.

6.3 Summary

We propose an evaluation metric that considers semantically similar, but distinct, temporal relations to evaluate automated systems that extract temporal information. Our metric also evaluates how well a system extracts events and temporal expressions. Additionally, this metric produces a single score, which could be used to identify the temporal awareness of a system. Our approach is intuitive and easy to implement. We implemented the metric using a Timegraph for handling temporal closure in the TimeML derived corpora, which makes the implementation scalable and computationally inexpensive.

Our proposed metric has been adopted to evaluate participants in TempEval-3. However, our metric does not reward a few implicit relations. To resolve this issue, we propose an updated solution, which rewards those implicit relations, but it does so very insignificantly. Future research in this area can explore this problem.

Next we propose the **evaluation of temporal information understanding** with temporal question-answering. Our proposal is motivated by the benefits inherent in the use of QA to assess the understanding of natural language. The benefits of using temporal QA for temporal information understanding are, (i) QA represents a task-focused way of evaluation, which is natural and simple for humans; (ii) it is much easier to create temporal questions (needed to evaluate temporal information by QA) than annotating new documents with temporal information (needed to evaluate by corpus). Hence, the evaluation by QA allows to create a larger test set more easily. (iii) moreover, QA also allows the evaluation of temporal understanding in other domains, which enables the testing of the generality of the systems.

We find that the existing human annotations are not complete enough to infer the implicit relations needed to answer all the questions. As a result, when evaluating by the corpus-based evaluation against the human annotation, we are comparing against something incomplete. QA based evaluation, however, does not entail the same problem.

Since humans ask questions about relevant information, QA scores would better capture the understanding of important temporal information as opposed to the corpus-based evaluation, in which all information is equally important for scoring. Our experiments show that the QA scores of two systems correlate to the results obtained by the corpus-based evaluation. However, the QA-based performance clarifies the system capabilities in the target application.

Future researchers can explore our QA evaluation to analyze the performance of automated temporal information understanding systems in new domains, such as the medical domain and the educational domains.

7 Improvements over the Existing Resources

In this chapter, we describe the contributions of this dissertation in improving the existing temporal resources.

At first we present the TRIOS-TimeBank corpus, an extended TimeBank corpus with additional semantic information. Next, we demonstrate algorithms to merge multiple temporal annotations. Finally, we describe our contributions to TempEval-3 Shared Task.

7.1 An Extension of the Existing Temporally Annotated Corpus and Annotation Scheme

Temporal annotation is hard task for humans, low inter-annotator agreement[1] for TimeBank corpus explains this very well. Since our system (section 4) extracts these information automatically and systematically, it can be used to find human errors. At the same time, our system uses semantic parsing and extracts other information that TimeBank does not include. With these

[1]http://www.timeml.org/site/timebank/documentation-1.2.html#iaa

benefits, we approached the task of extending the TimeBank corpus. Our first goal is to suggest missing TimeBank events and temporal expressions, i.e. events and temporal expressions that were missed by TimeBank annotators. Along with that we also suggest some additions to TimeML language by adding new event attribute (such as ontology type), some more SLINKs and also relations between events with their arguments, which we call RLINK (relation link). With our new suggestions we present the TRIOS-TimeBank corpus, an extended TimeBank corpus. In this section, we will describe our techniques very briefly. Our newly developed corpus is available online[2].

7.1.1 Suggesting New Events in TimeBank

The low inter-annotator agreement in TimeBank suggests that there should be some effort to refine TimeBank events. It is hard to automatically suggest that some annotated event in TimeBank is wrong; so we only **suggest new events that are missing in TimeBank**.

The TimeML (PCI$^+$03) specification suggests not to tag Generic interpretations, even though capturing them could be of use in question answering. By generics, they mean, events that are not positioned in time, or in relation to other temporally located events in the document. For example, they wouldn't annotate *use* and *travel* in the sentence: *Use of corporate jets for political travel is legal.*

It also suggests not to tag subordinate verbs that express events which are clearly temporally located, but whose complements are generics. For example, *He said Jews are prohibited from killing one another.* Even though the verb *said* is temporally located, it isn't tagged because its complement, *Jews are*

[2]TRIOS-TimeBank corpus is available online at: http://www.cs.rochester.edu/u/naushad/trios-timebank-corpus

prohibited from killing one another, is generic.

And finally an event nominalization that does not provide any extra information than the supplied verbs, are also not tagged.

Many of the extra events generated by TRIOS that are not in TimeBank fall into these categories. We made a decision to only suggest verbal events, so we do not have to worry about the last case. For verbal events, our task would be to keep the events that matches with TimeML specification.

There is one case where we decided to tag events even though they don't meet the TimeML specification. An example is, *He said the earth is round. They killed him.*. Here, *"the earth is round"* is generic by their definition. According to their scheme, *"said"* won't be annotated. But this saying event might explain rest of the story, i.e. in this case, why he was killed. If we don't annotate these kind of events, we are removing the information that we are interested in. So, we keep these kind of events in our event suggestion list, but will try to distinguish and eliminate other generic events.

The extra events that TRIOS finds can be categorized as follows: i. the result of wrong parse, ii. a generic event and iii. a legitimate event but missed by annotators.

Here are few examples that we think are legitimate events and also missed by TimeBank annotators:

An intense manhunt ***conducted*** *by the FBI and the bureau of alcohol, ...*

If Iraq ***chooses*** *a simple war of nerves and economic attrition, the Bush administration knows...*

American strategists are calculating, though, that the trade sanctions – enforced by an effective though perhaps undeclared naval blockade – will ***hold*** *tightly enough to* ***convince*** *Iraq that it will* ***lose*** *in the long run by simply standing pat.*

We are interested in suggesting these legitimate events and filter out generic and wrong events. The first level filtering is done by only keeping the events that are suggested as Verbs by both TRIPS parser and also Stanford POS tagger. Then the final filtering is done by classifying these extra events into "suggestion" , "generic" and "wrong" categories. To do this task, we implemented a MLN (markov logic network) classifier using *TheBeast* tool[3]. We generated the formulas for MLN from TRIOS generated event features. Flowchart for suggesting new TRIOS events is shown in Figure 7.1.

To analyze, we picked 40 TimeBank documents and annotated the extra TRIOS events with "suggestion", "generic" and "wrong" categories. We used 20 documents as test data and other 20 documents as training data. Since we did not have enough annotated data, we randomly picked some TimeBank events as "suggestion" instance and some non-verbal extra TRIOS events, which were filtered out in our first level filtering, as "wrong" instances. We added these new instances with our training data and then tested on our unseen 20 documents.

In these 20 documents test data, we had 90 extra events after our first level filtering, where we had 14 events which were result of the wrong parse, 41 generic events and 35 events that we think are legitimate events but missed by TimeBank that we want to suggest. The performance of classifying these categories are reported in Table 7.1.

In full TimeBank, we nominate 484 events as new event. From the system's performance, we expect that we extracted almost 90% of probable missing events and around 50% of these events are legitimate. These nominated new events are small in number. Hence, suggestions from this tool will help

[3]MLN Tool TheBeast: http://code.google.com/p/thebeast/ . All MLN classifier in this paper are implemented using TheBeast

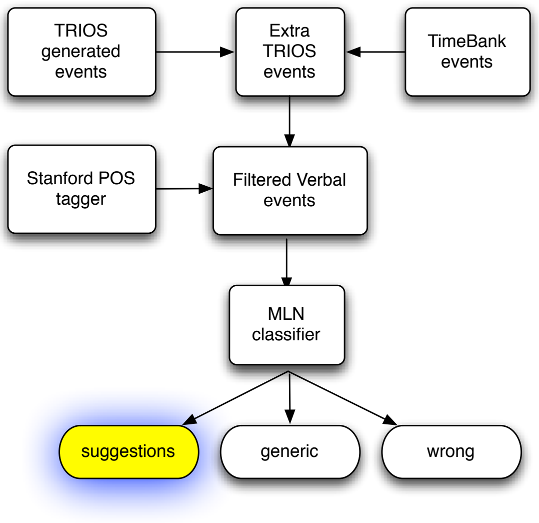

Figure 7.1: Flowchart to classify extra TRIOS events to "suggestion" , "generic" and "wrong" categories

significantly the annotator to add new events to TimeBank.

Category	# Instance	Precision	Recall
suggestion	35	50.00%	88.57 %
generic	41	70.00%	34.14%
wrong	14	87.57%	50.00%

Table 7.1: Performance on classification of extra TRIOS events into "suggestion", "generic" and "wrong" category

7.1.2 Adding New Event Attribute - *Ontology type*

TimeML comprises of event attributes, *class, tense, aspect, nf-morph, pos, modality, and polarity.* TRIOS system generates these attributes and also add the *ontology type* as event attribute, which is later used as feature for relation classification task.

Ontology type is the semantic type of word, particular word sense in the context, in the TRIPS ontology[4]. TimeML tries to capture the event information by very high level *class* or *pos*. Ontology type attribute will try to capture more fine grained information about the event, but in higher level than event word.

An example will explain the granularity better. Ontology type instances for "*fought*" in the sentence (example (4.1)) "*He fought in the war*" is FIGHTING. Few other words with ontology type FIGHTING would be: *contend, defend, and struggle*, i.e. these words with similar meaning will get the same ontology type, in this case FIGHTING.

TRIPS Ontology will be available for public use, so people can use the ontology for their system. It also has mapping to WordNet. So converting it to WordNet classes, someone can take benefit of this ont-type attribute.

TRIOS system generates event attributes from TRIPS parser output. For classifying the attribute *class*, we implemented a MLN classifier and used rest of the event attribute as features for classifier. In this classification problem, our system with TRIOS generated attribute performed with 77.3%[5] and the same system with TimeBank's event attribute performed with 77.47% accuracy. It is worth mentioning that TimeBank's inter-annotator agreement (IAG) on *class* is 77%. Comparing IAG and TimeBank's feature, our performance suggests that TRIOS generated features are equally good, which is an indication of ontology-type's performance as well.

[4]TRIPS ontology browser: http://www.cs.rochester.edu/ research/trips/lexicon/browse-ont-lex.html

[5]All performance for *class* identification are reported using 10-fold cross-validation.

7.1.3 Suggesting New Temporal Expressions in TimeBank

We extract temporal expressions from raw text by making a hybrid between a CRF-based engine and TRIPS extraction. For suggesting extra temporal expressions, we consider both systems, but do it slightly differently. We get the temporal expressions suggested by both systems and compare with TimeBank. Then we process the extra temporal expressions in a filtering step, which tries to extract the normalized value for the temporal expressions. If the normalization module gets a normalized value then we consider that temporal expression as suggestion. The flowchart for suggesting new temporal expressions is shown in Figure 7.2.

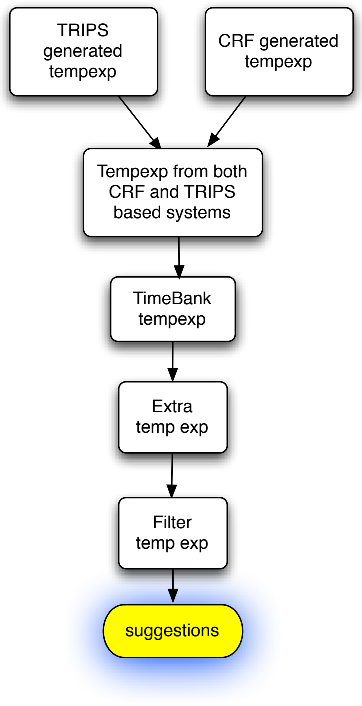

Figure 7.2: Suggesting new temporal expressions

The inter-annotator agreement for temporal expression identification is

96%, which means this is comparatively easy task for human annotators and we do not expect the annotators to miss many temporal expressions. On full TimeBank we suggested around 68 new temporal expressions, out of which, we found 50 (73.5%) temporal expressions to be legitimate. We added these new temporal expressions to the corpus. Some examples are shown below.

1. *At the end of the broadcast this evening, one more trip around Havana to see what it's been like since* ***the last time****.*

2. *And even terrorist groups that opposed Iraq in its war with Iran show signs of swinging behind Saddam Hussein* ***now*** *that he is in a confrontation with the U. S. ...*

3. *Turks feel they have special ties to the whole region, which they ruled for* ***hundreds of years*** *during the Ottoman Empire.*

4. *In* ***the first days*** *after President Bush announced the dispatching of U. S. troops, ...*

5. *Weisfield's, based in Seattle, Wash.,* ***currently*** *operates 87 specialty jewelry stores in nine states.*

6. ***Previously****, watch imports were denied such duty-free treatment.*

While the annotators did not miss any obvious dates, they missed some temporal expressions like *now*, *currently*, *last time*, *previously*, which are identified by our system and suggested as new temporal expressions. Such temporal expressions, although they have no specific temporal location as dates, helps to capture better temporal structure and are also annotated in TimeBank in general. Hence we want to suggest these new extra temporal expressions.

7.1.4 Adding Improved Relations in The Existing Annotation Scheme

Our next contribution is adding a richer set of relations to TimeML. TimeML captures the relations between different events with TLINK (temporal links), SLINK (subordinate link), and ALINK (aspectual link).

More SLINK instances: SLINKs or Subordinate Links are used for relations between two events. TimeML classifies SLINKs into *Modal, Factive, Counter-Factive, Evidential, Negative evidential and Conditional.* This classification leaves out many instances where two events are related to each other, i.e. one event is argument of another event. We try to capture all possible relations when one event is related to another event. In following three examples from TimeBank corpus, we make one event in each sentence bold and another underlined. The bold event is the core event and the underlined is the reference event and the relation type (relType) is noted in the bracket afterwards.

*(1) Integra, which owns and operates hotels, **said** that Hallwood Group Inc. has agreed to exercise any rights that aren't exercised by other shareholders. (Theme)*

*(2) "They have to **continue** to tighten their belts," said Craig Kloner, an analyst at Goldman, Sachs amp Co. (Purpose)*

We try to capture all these relations as SLINK and the relation type will be the semantic role (or thematic roles). The most common semantic roles (relation types) found in the corpus are shown in Table 7.2 along with comparison with equivalent semantic roles from VerbNet and Lirics.

There are other kind of SLINKs that we consider. Another instance from example (4.1) (*He fought in the war*) is:

```
<SLINK signal=IN eventInstanceID=V2 subordinatedEventInstance=V5
```

Our Role	VerbNet equivalent	Lirics equivalent	SLINK Count	RLINK Count
Agent	Agent, Actor	Agent	19	709
Theme	Stimulus, Theme	Theme	336	1137
Affected	Patient	Patient	13	92
Cause	Cause	Cause		49
Goal-as-Loc	Destination	finalLocation	47	
To-Loc	Recipient	Goal	46	
At-Loc	Location	Location	42	
In-Loc	Location	Location	28	
On	Location	Location	20	
Situated-In	Location?	Location?	39	
Purpose	–	Purpose	226	

Table 7.2: Most common relTypes used in SLINKs and RLINKs

```
relType=SITUATED-IN>
```

We also try to capture the signal (connectives, that connects two events). The problem in these cases is identifying the relation type (relType). We decided to use the ontology type of our signal (connective) as the relType for these kinds of extra SLINKs.

We suggest around 900 SLINKs to TimeBank corpus. Table 7.2 shows the statistics of most frequent SLINK types that we suggested.

New Relation Link, RLINK: Many researchers (Chambers et al., 2007 (CWJ07)), (Yoshikawa et al., 2009 (YRAM09)) showed that having dependency information improves the performance for extracting temporal relations. They tried to capture the dependency relation with dependency parsers like Stanford dependency parser. This gives a hint that capturing how other dependent words are connected with the event will enrich the information about the event.

We introduce new relation link, RLINK, to capture what other objects are related to the event (other than another event, which is captured with SLINKs), i.e. relation of event with its arguments.

In our initial example, for event FIGHT, we try to capture the information that the AGENT of that fighting event is HE, which is a PERSON. These relations give us information what are the arguments of an event and how they are connected.

```
<RLINK eventInstanceID=V2 ref-word=HE ref-ont-type=PERSON
      relType=AGENT>
```

We considered the thematic/semantic roles that described in Table 7.2. In TRIOS-TimeBank corpus we suggest around 2000 RLINKs. We showed the distribution in Table 7.2.

Another example of RLINK's importance could be explained with (Chambers and Jurafsky, 2008) (CJ08). They learned narrative event chains considering the idea of protagonist (central actor). They are basically considering the events performed by the same agent. We are trying to capture the agent and other different thematic roles (or semantic roles) using RLINK, which would help many other applications like Chambers and Jurafsky (2008).

One argument against adding RLINK is, these information could be annotated with other layers, such as syntactic and semantic layers. We agree that this information could be added using other layers, however, for building a complete temporally aware system we would need these information. Hence to build a complete temporally annotated corpus it is better to have this information as well. This would benefit advanced applications, such as question answering, summarization, etc.

7.2 Merging Multiple Temporal Annotations

In another effort, we explore whether or not the correctness of temporal annotations can be improved by merging multiple annotations using automated algorithms. Our hypothesis is that merging annotations may benefit in many different ways, such as:

1. Improving over the performance of individual automated systems in various ways.
    - Building a high recall system: If we take different systems, one system can fail to extract some elements and the other system can have better coverage for those particular elements. By merging, the missing elements of one system can be covered by another; as a result we can achieve a high recall system.
    - Building a high precision system: By taking votes from multiple systems, we can ignore some low-confidence information to build a high precision system.
    - Building an overall better system (balanced precision and recall): Increasing the recall by combining all system output will decrease the precision and increasing the precision by ignoring low-confidence information will decrease the recall. However, a balanced combination may help to improve performance in general.
2. Automatically or semi-automatically annotating temporal corpora.
    - Merging different human annotations: When we have different human annotations, it is hard to merge them manually. An automated merger can be used to merge the output easily.

- Increasing human annotation coverage: Temporal annotation is a hard task even for humans[6]. Human annotators can often miss some useful information. By merging human annotation with state-of-the-art automated annotations, we can increase the coverage.
- Annotating large corpora: Finally, with merging algorithms that improve over state-of-the-art systems, we can automatically annotate a large corpus that could not be annotated by manual means. In addition, if we use high recall merging, it can be easily improved with human review mainly by removing information.

The underlying difficulty of merging annotations is that it cannot be ensured that the result will be better than the individual annotations. When merging annotations it is unknown whether or not an annotated element is correct. It is crucial therefore that the merging algorithms use some heuristics to merge different elements.

In this section, we propose and develop algorithms to merge multiple temporal annotations. To get a better merging, the algorithms consider the system/annotator performance and the temporal annotation particularities (e.g., temporal relation consistency). We evaluate the performance of the proposed merging algorithms and discuss how merging can be used to achieve the benefits introduced above.

This section is structured as follows. First, we describe different merging techniques employed in various natural language processing (NLP) applications. Next, we present our merging algorithms. Finally, we report and discuss our experimental results and draw the conclusions reached.

[6]Find the inter-annotator agreement of TimeBank corpus at: `http://timeml.org/site/timebank/documentation-1.2.html#iaa`

7.2.1 Related work

In NLP, a range of merging techniques have been employed for improving different tasks such as part-of-speech tagging or syntactic parsing.

Sjobergh (Sj03) and Brill & Wu (BW98) combined different POS-taggers. Reidel et al. (RMS$^+$11) combined bio-molecular event extraction systems. Swift et al. (SAG04) combined parsing output from different sources.

The techniques used in these papers are:

- Simple voting: this is the most common technique (Sj03; BW98). This is equivalent to weighted voting using same weight for all the inputs.
- Weighted voting: this normally involves giving higher weights to better systems. Sometimes the systems also output confidence score for each individual prediction. These scores for predictions have been also used as weights for voting (Sj03).
- Stacked merging: there are two main approaches for stacked merging: (i) using the output of one system as input feature of another system (SAG04; RMS$^+$11); and (ii) taking all system outputs and training a new model with these outputs (Sj03).

In our merging, we exploit simple and weighted voting techniques with specific heuristics to merge temporal annotations. Due to the complexity of temporal annotation, we did not explore stacked merging.

7.2.2 Merging Temporal Annotations

Our goal is to take a set of TimeML annotations of the same text from different sources – either automated systems, human annotators, or both – and automatically obtain a merged annotation. In particular, we aim to get a merged annotation which is better than individual annotations in terms of correctness, i.e., get the correctly annotated elements of the annotations and avoid the incorrect ones. Since it is unknown which elements of each annotation are correct, the problem is to find heuristics useful, in particular, for merging temporal annotations.

We present two approaches to this problem: bottom-up and top-down. Each approach is defined based on how the elements in the annotation are merged. Bottom-up first merges the entities (events and temporal expressions) from all the sources and then merges the temporal relations from the sources that contain those entities. Top-down merges the temporal relations first and then based on the merged temporal relations, the entities.

In both approaches, we use voting to decide which elements to keep in the merged annotation. In particular, we use weighted voting to be able to give higher weights to better sources to ensure that a priori better annotations are preferred. We only include an element in the merged annotation if its voting is above a threshold (merging threshold). The thresholds, for entities and relations, are customizable in our algorithms, i.e., these can be modified to setup different merging configurations (e.g., threshold 0% is equivalent to unifying all annotations or threshold 50% to selecting by majority).

Before running the merging algorithms we normalize the annotated elements. TimeML annotation output from different sources may have different elements (e.g., annotation A has an event which is missed in annotation B). In order to merge annotations we first need to identify which elements are the same in different annotations. We therefore developed an algorithm which gives the same id to the same elements from different sources.

Entity normalization: We consider entities to be the same when there is a partial overlapping of the text between entities of two different sources, e.g. if one source identifies Sunday and another source identifies Sunday morning, we will identify both of these as same entity. In order to carry out this task, the algorithm uses the entity position in text.

Relation normalization: A temporal relation links two entities with a relation type. We give the same id to the relations from different sources that relate the same two entities (i.e., with the same id). If the entities are related in different order, e.g. one source has e1 e2 r1 and another source has e2 e1 reverse(r1), then we will make both e1 e2 r1.

Normalized TimeML annotations from different sources are the input of our merging algorithms.

Bottom-up merging

Merging entities: In bottom-up, entities are merged first. We consider the weighted votes from all annotations to decide if an entity is kept or not. We keep the entity if the voting is above a predefined threshold. The approach to select the entities and their main attribute values[7] is explained in Algorithm 1. In the case of multi-token entities, the algorithm also selects the extent of the entities – tokens included in the entity. Since obtaining the correct attribute value often requires considering the correct entity extent, for each entity, after selecting which attribute value to keep in the merged annotation,the algorithm checks which sources suggested that attribute value and selects the extent and

[7]Main attribute is class for events and value for timex.

other attributes by voting only from these sources.

```
// $sources$ = all sources for merging
// $entities$= all entities from all sources
// w[s] is the weight for source s

foreach $entity\ e \in entities$ do
    foreach $source\ s \in sources$ do
        if $e \in s$ then
            $entity\_weight[e]$ += $w[s]$
$merged\_entities = \{\}$
foreach $entity\ e \in entities$ do
    foreach $source\ s \in sources$ do
        if $e \in s$ and $entity\_weight[e] > threshold$
        and $e \notin merged\_entities$ then
            $value$ = get_attribute_value($e$, $attribute$, $s$)
            $attribute\_value\_weight[e][value]$ += $w[s]$ $merged\_entities$.append($e$)
```

Algorithm 1: Calculating weight for entities

Merging relations: After getting the merged entities from Algorithm 1, we merge, from all sources, the relations which link those entities. In order to merge them we distinguish two matching levels: **triples** and **doubles**. We define relation triple as entity1 entity2 relation_type (complete match) and relation double as entity1 entity2 (only entities match). We use triples and doubles to calculate the weighted voting as described in Algorithm 2. Doubles are useful because they represent when two entities are related, but triples are much more important because they represent how they are related. In the algorithm, to sort relations in terms of triple weights first and then double weights, given n sources, we multiply triple weights by (n+1). This gives more weight to one triple than n doubles.

```
// $n$ = number of sources
// $relations$ = relations whose entities are contained
//                 in merged entities (algorithm 1)
// get_double_from_triple(triple $t$) = remove relation category from t
// e.g., get_double_from_triple($e1, e2, BEFORE$) = $e1, e2$

foreach triple $t \in relations$ do
    foreach source $s \in sources$ do
        if $t \in s$ then
            $weight\_triple[t]$ += $w[s]$ * ($n$+1)
            $d$ = get_double_from_triple($t$)
            $weight\_double[d]$ += $w[s]$
foreach triple $t \in relations$ do
    $d$ = get_double_from_triple($t$)
    $relation\_weight[t] = weight\_triple[t] + weight\_double[d]$
$merged\_relations = \{\}$
foreach triple $t \in relations$ do
    if $relation\_weight[t] > threshold$ and $consistent(t)$ then
        merged_relations.append($t$)
```

Algorithm 2: Calculating the weight for relations

Example (1) illustrates how the algorithm merges the relations from three sources (S1, S2, S3) with different weights.

(1) w[S1] =4, w[S2]=3, w[S3] = 2
    triples[S1]: {e1, e2, BEFORE}, {e2, t1, AFTER}
    triples[S2]: {e1, e2, BEFORE}, {e2, t1, ENDS}
    triples[S3]: {e1, e2, AFTER}, {e5, e6, AFTER}
    relation_weight[{e1, e2, BEFORE}] =
        (w[S1]+ w[S2])*4 + w[S1] + w[S2] + w[S3] = 37
    relation_weight[{e2, t1, AFTER}] =
        w[S1]*4 + w[S1]+w[S2] = 23
    relation_weight[{e2, t1, ENDS}] =
        w[S2]*4 + w[S1]+ w[S2] = 19
    relation_weight[{e1, e2, AFTER}] =
        w[S3]*4 + w[S1] + w[S2] + w[S3] = 17

relation_weight[{e5, e6, AFTER}] = w[S3]*4 + w[S3] = 10

With Algorithm 2, we will get a sorted list of relations, i.e. highly voted relations on top. This allows removing the less voted inconsistent ones. We assume that temporal annotations should be consistent and thus also the merging output.[8]

To check temporal consistency we need to calculate the temporal closure. This is a computationally complex task. Our merging algorithms check temporal consistency using Timegraph (section 5.3.1), which creates a temporal structure in which temporal consistency can be efficiently checked.

The less voted relations that are inconsistent with the already added relations (more voted) are eliminated from the list. Finally, the consistent merged relations that are above the established merging threshold are included in the merged output.

Top-down merging

The top-down merging approach first merges the temporal relations. We apply the same algorithm described for bottom-up merging in Algorithm 2, but instead of starting with relations of merged entities, we start with all relations from all sources. For merging entities, we by-default include the entities that participate in the merged relations. For the rest of the entities, we apply the entity merging approach described in Algorithm 1.

[8]Even knowing that some gold standard annotations are erroneously inconsistent in terms of closure.

7.2.3 Evaluation and Discussion

In this section, at first, we describe the corpora, the annotation sources, and the evaluation metrics that we used for our experiments. Finally, we present and discuss the results.

Corpora

We consider TimeBank 1.2 and AQUAINT[9], which are the available TimeML compliant corpora. We use both corpora together and we perform 10-fold cross validation for all the experiments, i.e., we report the mean of 10 complementary experiments using 90% training and 10% test.

Annotations being Merged

We take as annotation sources three systems from TempEval-2, which participated in all the tasks and obtained the best scores. These are TIPSem and TIPSemB (LSNC10); and TRIOS (described in section 4). In order to obtain TimeML annotations from raw text with these systems, we extended them in two respects: (i) we added a module to identify which entities to pair for temporal relations; and (ii) we trained the relation categorization model on the full set of TimeML relations – as annotated in TimeBank and AQUAINT– instead of the reduced set used in TempEval-2.

Evaluation Metric

The evaluations can be divided into entity evaluation (for events and temporal expressions) and relation evaluation.

To evaluate the entity performance, we consider the entity based evaluation (equation 3.1 – detail in section 3.3.1).

[9] `http://timeml.org/site/timebank/`

For attributes, we report the attribute recall (equation 3.6). In TempEval-2, the attribute performance was reported as attribute accuracy – calculated as the matching attributes out of matching reference and system entities. If an annotation matched only only one entity and gets its attribute correct, then it gets 100% accuracy; just as one annotation that has matched all the entities and attributes. This makes comparing attribute performance between annotations difficult. In order to make comparison easier, we used the attribute recall – calculated as the number of matching attributes and entities out of total reference entities. Attribute recall is equivalent to the multiplication of entity recall and attribute accuracy.

To evaluate the performance of temporal relations, we proposed an evaluation metric in section 6.1.3. Our metric has been adopted to evaluate the participants of TempEval-3, identifying temporal relations. This metric captures the temporal awareness of an annotation in terms of Precision, Recall and F score. Unlike TempEval-2 relation score, where only relation categorization is evaluated, this metric evaluates how well pairs of entities are identified, how well the relations are categorized, and how well the events and temporal expressions are extracted.

Results and Discussions

The objective of this evaluation is to analyze whether or not the application of our algorithms leads to a merged temporal annotation which is better than the individual annotations used to obtain it.

We evaluate various voting configurations for merging annotated elements. Each configuration sets a voting threshold which has to be reached by an element to be kept in the merged annotation. In some experiments we use weighted voting – not all the annotation sources have the same weight. In order to assign weights to individual annotations, we measured their temporal awareness F score in a separated subset of the data (TempEval-2 test documents in TimeBank). When it is needed, we use their performance as weight, i.e., 32% for TIPSem, 29% for TIPSemB, and 27% for TRIOS.

As baseline configurations, we consider union and intersection. In union, all the annotations have the same weight (33%) and the threshold is 0%, i.e., only 1 vote is required and then all the elements are kept. In intersection, all the annotations have the same weight (33%) and the threshold is 99.9%, i.e., all the votes are needed to keep an element.

We suggest three merging configurations. Firstly, majority (MAJ), where we give same weight to all annotations (33% – simple voting); and the threshold is set to 50%, i.e., majority is required (2 votes). Secondly, performance (PERF), where we give each annotation expected correctness (system performance) as the weight and the threshold is set to the lowest weight (27%). In this configuration, we give preference to the better annotations as regards their performance. Finally, balanced (BLNC), where we give the best annotation its performance as weight (32%) and the other annotations a lower and equal weight (27%); the threshold here is also the lowest weight (27%). In this configuration we give preference to the best annotation and for the rest we require at least 2 votes to include their elements in the merged annotation.

In PERF and BLNC, we expect a better balance between precision and recall because these will include the majority of the elements of the better annotation and only those elements of the worst annotations that get some support (more than one vote).

Below we report on the results for the bottom-up approach for event, timex, and relation merging. For each of the cases, we compare the performance of our different merging approaches against baselines, and then we compare the results of the individual annotations (systems) against the PERF approach, which obtained the best average results.

EVENT: For event extraction, our PERF configuration obtained the best results, closely followed by BLNC. All our merging configurations are also better than our baselines. Performance in F1 is reported in Table 7.3.

	union	**intersection**	**MAJ**	**BLNC**	**PERF**
F1	84.9	64.6	85.1	86.7	**86.9**

Table 7.3: Merging performance for event extraction

Furthermore both PERF and BLNC outperform all individual annotations that we merge. Performance in F1 is reported in Table 7.4.

	TIPSem	**TIPSemB**	**TRIOS**	**PERF**
F1	86.6	85.5	75.1	**86.9**

Table 7.4: Systems vs. PERF merging for event extraction

The mean positive difference in the 10-folds between PERF and TIPSem was 0.10 with a mean standard deviation of 0.07, which supports our hypothesis that merged annotation improves over individual annotations.

TIMEX: For timex extraction, PERF performed the best among our configurations, closely followed by BLNC. However, the union baseline obtained the best results for timex extraction task. This is due to the fact that, in general, union obtains high recall and it can get high F1 if it can maintain a high precision at the same time. For timex extraction, individual systems have high precision and then union performs very well. The F1 of the baselines and our merging configurations are reported in Table 7.5 and a further details for union are discussed in the next subsection in Table 7.10.

Both PERF and BLNC outperform all individual systems that we merge. Performance in F1 is reported in Table 7.6.

	union	**intersection**	**MAJ**	**BLNC**	**PERF**
F1	**91.6**	60.6	86.1	89.4	89.5

Table 7.5: Merging performance for timex extraction

	TIPSem	**TIPSemB**	**TRIOS**	**PERF**
F1	88.3	86.1	86.0	**89.5**

Table 7.6: Systems vs. PERF merging for timex extraction

The mean positive difference between PERF and TIPSem was 0.10 with a mean standard deviation of 0.08, which supports our hypothesis that merging improves also in timex annotation.

RELATIONS: For relations (temporal awareness), our BLNC configuration performed the best. MAJ performed better than PERF here, which we will explain in detail later. All our merging configurations are better than our baselines in temporal awareness. Performance in F1 is reported in Table 7.7.

	union	**intersection**	**MAJ**	**BLNC**	**PERF**
F1	29.35	16.98	30.90	**31.38**	30.65

Table 7.7: Merging performance for temporal awareness

All our merging configurations outperform all the individual annotations that we merge. Performance in F1 is reported in Table 7.8.

For temporal awareness, the mean positive difference in the 10-fold between the best system (TIPSem) and PERF was 0.15 with a mean standard deviation of 0.03.

DETAILED EXPERIMENTS: We also report the detailed scores to better analyze the merging results.

EVENT: We detail the event performance of best individual system (TIPSem) and average best merging configurations (BLNC, PERF). For event attributes

	TIPSem	**TIPSemB**	**TRIOS**	**PERF**
F1	29.59	28.41	24.75	**30.65**

Table 7.8: Systems vs. PERF merging for temporal awareness

(class recall), both our best merging configurations outperform TIPSem. Table 7.9 includes event precision (P), recall (R), fscore (F1), class accuracy (class ac) and class recall (class R).

	P	**R**	**F1**	**class accuracy**	**class Recall**
TIPSem	87.5	85.7	86.6	82.7	70.8
BLNC	87.0	86.5	86.7	82.8	71.6
PERF	85.8	88.0	86.9	82.7	72.8

Table 7.9: Detail event performance for best individual system and best merging configurations

TIMEX: In this case it is worth examining the UNION baseline in addition to the best individual system (TIPSem) and best merging configurations (PERF, BLNC). In timex attribute (value recall), both our best merging configurations outperform the best individual system; and UNION performs the best as it is shown in Table 7.10.

	P	**R**	**F1**	**class accuracy**	**class Recall**
TIPSem	94.6	82.8	88.3	70.5	58.4
PERF	93.9	85.7	89.5	71.7	61.4
BLNC	94.7	84.7	89.4	72.0	61.0
UNION	91.0	92.4	91.6	70.2	64.9

Table 7.10: Detail timex performance for best individual system and best merging configurations

Since the precision of the annotations being merged is high the key point for timex is improving recall. The best way to do so is by a union merging.

RELATIONS: Table 7.11 shows the detail temporal awareness performance of best individual system (TIPSem) and all the merging configurations.

	P	**R**	**F1**
TIPSem	26.46	33.62	29.59
BLNC	28.68	34.71	31.38
PERF	26.55	36.35	30.65
MAJ	35.27	27.54	30.90

Table 7.11: Detail temporal awareness performance for best individual system and best merging configurations

Table 7.7 showed previously that MAJ performs the second best. But we can get the insight from Table 7.11 that the improvement was due to high precision. Table 7.11 also shows the strengths of each of our configurations. MAJ has high precision, PERF has high recall and BLNC is a good balance between precision and recall. Depending on the need, we can use the best configuration for our purpose to get the best out of our merging algorithms.

ADDITIONAL EXPERIMENTS: We did few additional experiments to make a better analysis.

Experiment 1: We checked if merging only two systems, instead of three, still improves over individual systems. We experimented the merging of two systems in PERF configuration, which is equivalent to BLNC when merging only two systems. The results are reported in Table 7.12.

	TIPSemB-TRIOS	**TIPSem-TIPSemB**	**TIPSem-TRIOS**
F1	30.12	31.15	31.33

Table 7.12: Performance of merging two systems

In all two-system combinations we have found that our merged annotations outperform the individual best system. TIPSem and TIPSemB are the two best individual systems. It could be expected that their combination would lead to the best result. However, TIPSem-TRIOS combination performed better. This is because TIPSem and TRIOS are the most different systems and their annotations are more complementary (better for merging), while TIPSemB is a variant of TIPSem and their annotations are more similar.

Experiment 2: We repeated all the previous experiments using top-down approach, but we did not find any notable differences for any task between top-down and bottom-up. The comparison temporal awareness is reported in Table 7.13.

	MAJ	**BLNC**	**PERF**
Bottom-up	30.9	31.38	30.65
Top-down	31.23	31.18	30.52

Table 7.13: Comparison between top-down and bottom-up

Experiment 3: For all the experiments, we only considered the relations that are consistent, i.e. maintains the closure property. We evaluated our configurations keeping all the relations, i.e. including inconsistent relations. The performance is reported in Table 7.14.

	MAJ	**BLNC**	**PERF**
without inconsistent	28.68	34.71	31.38
with inconsistent	27.09	35.25	30.60

Table 7.14: Performance with/without inconsistent relations for BLNC

We found a slightly better recall when including inconsistent relations since we are considering more relations, but overall performance decreases because of lower precision.

7.2.4 Summary

We showed that our merging algorithms improve the performance over individual annotations. We also showed that using different merging configurations, we can achieve high recall, high precision and balanced annotations.

7.3 TempEval-3

In this section we describe the contributions of this thesis for TempEval-3[10] (ULA$^+$12). We start by describing the TempEval-3 shared task and then we explain our preparations for the task.

7.3.1 TempEval-3 Shared Task

Temporal annotation is a time-consuming task for humans, which has limited the size of annotated data in the previous editions of TempEval. Current systems, however, are performing close to the inter-annotator reliability, which suggests that a larger corpus could be built from the automatically annotated data with minor human reviews. We want to explore whether there is value in adding an automatically created large silver standard data to a small handcrafted gold standard data. An auto-annotated larger corpus could be more useful than a small hand annotated corpus for some tasks. With this in mind, we propose the upcoming temporal evaluation shared task – TempEval-3.

TempEval-3 is a follow-up of TempEval 1 and 2. TempEval-3 is different from its predecessors in a few respects: (i) size of the corpus: the dataset will have about $\frac{1}{2}$ million word silver standard data, and about 100K word gold standard data for training, compared to around 50K word data used in

[10] `http://www.cs.york.ac.uk/semeval-2013/task1/`

TempEval 1 and 2; (ii) temporal relation task: the temporal relation classification tasks will be performed from raw text, i.e. participants need to first extract events and temporal expressions, determine which ones to link, and then obtain the relation types; (iii) temporal relation types: the full set of temporal relations in TimeML will be used, rather than the reduced set used in earlier TempEvals; (iv) annotation: most of the corpus will be automatically annotated by the state-of-the-art systems from TempEval-2, a portion of the corpus, including the test dataset, will be human reviewed; and (v) evaluation: we will report a temporal awareness score to evaluate temporal relations, which would help to rank systems with a single score.

7.3.2 Temporal Evaluation Shared Task 2013 – TempEval-3 Tasks

The proposed tasks for TempEval-3 are:

- **Task A (Temporal expression extraction and normalization)**: Determine the extent of the time expressions in text as defined by TimeML's TIMEX3 tag. In addition, determine the value of the features TYPE and VALUE. TYPE can be time, date, duration, and set; and VALUE is a normalized value as defined by TIMEX2 and TIMEX3 standards.

- **Task B (Event extraction)**: Determine the extent of the events in a text as defined by TimeML's EVENT tag. In addition, determine the value of the features CLASS, TENSE, ASPECT, POLARITY and MODALITY, and also identify if the event is a main event.

- **Task C (Identifying temporal relations)**: Identify the pairs of temporal entities (events or temporal expressions) that have temporal relations, and classify those relations. Possible pairs of entities that can have temporal relations are: (i) main events of consecutive sentences, (ii) pairs of events in the same sentence, (iii) event and temporal expressions in the same sentence, and (iv) event and document creation time. Now the participating systems need to identify the entities that need to be paired. The relation labels will be the same as the original TimeML's relations: *before, after, includes, is-included, during, simultaneous, immediately after, immediately before, identity, begins, ends, begun-by and ended-by.*

Participants may choose to do task A, B, or C. Choosing task C entails doing task A and B. However, a participant may perform only task C by applying available tools to carry out tasks A and B.

7.3.3 TempEval-3 Datasets

For TempEval-3, we reviewed and modified the existing corpora, and released the new corpora.

Reviewing the Existing Corpora

We review the existing TimeBank (PHS+03), TempEval-1 (VGS+07), TempEval-2 (VSCP10) and AQUAINT[11] for TempEval-3. TimeBank, TempEval-1 and TempEval-2 had the same documents, but different relation types and sometimes different set of events. We will refer to this corpus as TimeBank.

For both TimeBank and AQUAINT, we (i) cleaned up the formatting of all files making it easy to review and read, (ii) made all the files XML

[11] `http://timeml.org/site/timebank/`

and TimeML schema compatible, (iii) added the missing events and temporal expressions. In AQUAINT, we added the temporal relations between event and DCT (document creation time), which were missing for many documents in that corpus. In TimeBank, we (i) borrowed the events from the TempEval-2 corpus and (ii) borrowed the full-set of temporal relations from TimeBank corpus.

Automatically Creating New Large Corpora

We collected the $\frac{1}{2}$ million word corpus from Gigaword corpus[12]. We automatically annotated this corpus with TIPSem, TIPSem-B (LSNC10) and TRIOS (described in section 4). These systems were retrained on the TimeBank and AQUAINT corpora to generate the original TimeML temporal relation set. We then merged these three state-of-the-art system outputs using our merging algorithm (section 7.2). In our selected merging configuration, all entities and relations suggested by the best system (TIPSem) are added in the merged output. Suggestions from other systems (TRIOS and TIPSem-B) are added in the merged output, only if they are supported by another system. The weights considered in our selected configuration are: TIPSem 0.36, TIPSemB 0.32, and TRIOS 0.32. This automatically created corpus is referred to as silver data. A portion of the silver data is in the process of human revision to release it as an additional gold training data.

Our released corpora will consist of the sections described in Table 7.15.

* revision in process.

The participants and future researchers can explore the benefits of both – large automatically temporally annotated corpora (silver data) and small human annotated/reviewed temporal annotated corpora (gold data) – with

[12]`http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2011T07`

Corpus	Number of words	Purpose	Standard
TimeBank	61418	Training	Gold
AQUAINT	33973	Training	Gold
TempEval-3 Silver	666309	Training	Silver
TempEval-3 Gold	*20000	Training	Gold
TempEVal-3 Evaluation	*20000	Evaluation	Gold

Table 7.15: Available corpus released for TempEval-3

our TempEval-3 release.

7.3.4 Evaluating Participating Systems

To evaluate the entity performance on tasks A and B, we will consider the entity based evaluation (equation 3.1 – detail in section 3.3.1). For the attributes, we will report the attribute F score (equation 3.6 and 3.7).

The evaluation on task C will be incorporated from our proposed evaluation metric (section 6.1.3). Our metric uses temporal closures to reward relations that are equivalent but distinct, and then finds precision and recall. We showed that our F-score represents the overall temporal awareness of a system (i.e. it evaluates how a system extracts events, temporal expressions and identifies temporal relations).

7.4 Summary

This chapter describe the contribution of this thesis in improving the existing temporal resources. At first we present the TRIOS-TimeBank corpus, an extended TimeBank corpus with additional semantic information. We suggest additional TimeBank events, propose an extension to the TimeML language with richer event features, and event relations, all of which we generate using semantic parsing. The TRIOS-TimeBank corpus, with newly added events, event feature and relations, is available to the community for further research on temporal reasoning.

Next, we present algorithms to merge multiple temporal annotations. We discover that our merged annotations have an improved performance over that of the individual ones. We also demonstrate that by using different merging configurations, we can achieve high recall, high precision and balanced annotations.

In the experiments, we merged three state-of-the-art system annotations, and our merged annotation perform better than the state-of-the-art system annotations. The improvement found in the merged annotation is not significant but the annotations are always improved. Since we expect an improvement, we conclude that the merging is worthy even if it only slightly improves the annotations.

Finally, we describe TempEval-3 Shared Task and its different components. This dissertation contribute to TempEval-3 Shared Task by defining the task descriptions, by reviewing the existing corpora, by automatically creating the new large corpora, and by proposing the evaluation metric to evaluate the participants.

8 Conclusion

This chapter describes the contributions of this thesis and the possible future directions of this research work.

8.1 Contributions

This thesis focuses on understanding the temporal information in natural language. This involves automatic extraction, interpretation and reasoning about the temporal aspects of language. Such capabilities are key to many advanced natural language processing (NLP) applications, such as question answering, information extraction, document summarization and dialog systems. These techniques can be applied in news, medical, history and other domains.

The research on temporal information understanding evolved from the rationalist formal strategies (Chapter 2), proposed by the Linguistics and the Artificial Intelligence community, to the empiricist corpus-based strategy (Chapter 3) put forth by the Corpus Linguistics community. Having been influenced by both approaches, we proposed in this dissertation a hybrid system (in Chapter 4) to automatically extract temporal information from raw texts by extracting the events, the temporal expressions and identifying the temporal relations between them.

Our system for **extracting the temporal information** uses a combination of deep semantic parsing and machine learning classifiers. We compared our system with the existing systems performing the same task on TimeBank (PHS$^+$03) and TempEval (VGS$^+$07), (VSCP10) corpora. Our system outperforms or performs comparably with the existing systems.

We utilized our system that automatically extracts temporal information to **build a temporal structure** (section 5.3) using a Timegraph (section 5.3.1). Our temporal structure can answer how two entities are related to each other by making temporal inferences even when the relations are not explicitly mentioned. Finally, using the temporal structure, we presented a **question-answering system** (section 5.4), capable of temporal reasoning. Since our system is generic, it can be used to answer temporal questions about any document annotated in TimeML.

Next, to evaluate automated systems extracting temporal information, we proposed a new **temporal evaluation metric** (section 6.1) that considers semantically similar, but distinct, temporal relations and consequently gives a single score that could be used to identify the overall temporal awareness of a system. Our approach is intuitive and easy to implement. We implemented the metric using a Timegraph for handling temporal closure in the TimeML derived corpora, which makes the implementation scalable and computationally inexpensive. Our proposed metric has been adopted to evaluate the participants in TempEval-3.

Then we proposed the **evaluation of temporal information understanding** (section 6.2) with temporal question-answering. Our proposal is motivated by the benefits inherent in the use of QA to assess the understanding of natural language. The benefits of using temporal QA for temporal information understanding are: (i) QA represents a task-focused way of evaluation, which is natural and simple for humans; (ii) it is much easier to create temporal questions (needed to evaluate temporal information by QA) than annotating new documents with temporal information (needed to evaluate by corpus). Hence, the evaluation by QA allows creation of a larger test set more easily. (iii) moreover, QA allows the evaluation of temporal understanding in other domains as well, which enables the testing of the generality of the systems.

Next, we described the **contribution of this thesis in improving the existing temporal resources** (Chapter 7). At first, we presented the TRIOS-TimeBank corpus, an extended TimeBank corpus with additional events. We also proposed an extension to TimeML language with a richer event feature and event relations, all of which we generated with the help of deep understanding of text using semantic parsing. This resource, the TRIOS-TimeBank corpus, with the newly added events, the event feature and relations, is available to the community for further research on temporal reasoning.

Then we presented algorithms to merge multiple temporal annotations. We showed that our merged annotations have an improved performance over that of the individual ones. We also demonstrated that by using different merging configurations, we can achieve high recall, high precision and balanced annotations.

Finally, we described the TempEval-3 Shared Task and its different components. This dissertation contributed to TempEval-3 Shared Task by defining the task descriptions, by reviewing the existing corpora, by automatically creating the new large corpora, and by proposing the evaluation metric to evaluate the participants.

8.2 Future Work

8.2.1 Temporal Visualization or Timeline – Making information accessible

This dissertation presented the systems to extract temporal information and to answer temporal questions. The next step would be to create the temporal visualization or the timeline of a document to quickly sketch a summary of the document. This visualization to create a summary document could benefit a diverse group of people with reading difficulties – e.g. people whose first language is not English, individuals with low literacy, children, older people – and even those who want to quickly skim the document. We present a diagram in Figure 8.1 that we suggest for temporal visualization. However, this diagram mainly visualizes *before* and *after* relations. It becomes very complicated when all the relations (Table 2.3) are considered.

There is existing research on temporal visualization, e.g. Time-Surfer by Llorens et al. (2011) (LSNG11) and TBox by Verhagen (2007) (Ver07). However, these existing temporal visualizations are not detailed enough. On the other hand, visualizing fine grained temporal relations for all pairs of relations together is a big challenge due to the excessive amount of information, and finding a good balance of granularity is another difficult task. Future researchers can explore this area.

To make the visualization more appealing and more useful to a wider audience, illustrations can be added in the visualization as well (UBA10b) (UBA10a). Our existing work on Multimodal Summarization (UBA11) can be used to illustrate the complex sentences in a simplified form as an entity node, and the temporal visualization can show how all entity nodes are related to each other in terms of time. We present a diagram in Figure 8.2 that we

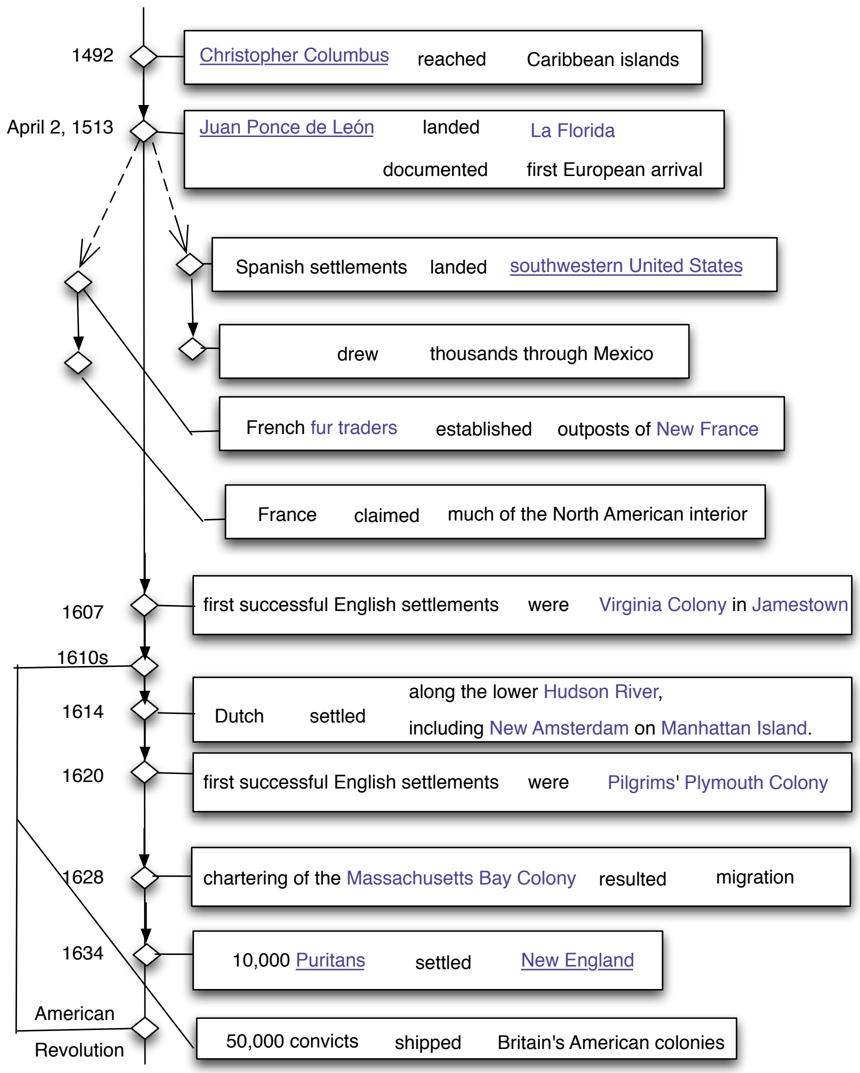
1492
Christopher Columbus reached Caribbean islands
April 2, 1513
Juan Ponce de León landed La Florida
documented first European arrival
Spanish settlements landed southwestern United States
drew thousands through Mexico
French fur traders established outposts of New France
France claimed much of the North American interior
1607
first successful English settlements were Virginia Colony in Jamestown
1610s
1614
Dutch settled along the lower Hudson River, including New Amsterdam on Manhattan Island.
1620
first successful English settlements were Pilgrims' Plymouth Colony
1628
chartering of the Massachusetts Bay Colony resulted migration
1634
10,000 Puritans settled New England
American Revolution
50,000 convicts shipped Britain's American colonies

Figure 8.1: Temporal Visualization of a section of a Wikipedia article on Christopher Columbus

suggest for temporal visualization with illustration.

A visualization with illustration also bears similar challenges due to the excessive information, and due to the difficulty in finding a good balance in granularity. One solution is to start with the minimum information, such as existing temporal visualization techniques (LSNG11), and then deliver more detailed information on demand.

8.2.2 Temporal Summary, Visualization and QA System

In this dissertation we have presented the ground work for building advanced NLP applications with temporal information processing. Future researchers can extend the work to build an application that can produce a temporal summary in text and a visualization at the same time for a document with the temporal QA capability. This combined application would be very useful for many people, such as doctors to understand and inquire about patients' historical records in medical NLP domains, for students/children to understand and learn history in education domains, for people with reading disabilities to follow news articles, and for many others as well.

8.2.3 Future Temporal Evaluation Shared Task - TempEval

TempEval has been the driving force behind the temporal information processing community. The first TempEval (VGS$^+$07) provided participants with all entity attributes and evaluated participants with the temporal relation classification task. The next TempEval (VSCP10) had different subtasks to evaluate participants on entity extraction and relation classification tasks. The upcom-

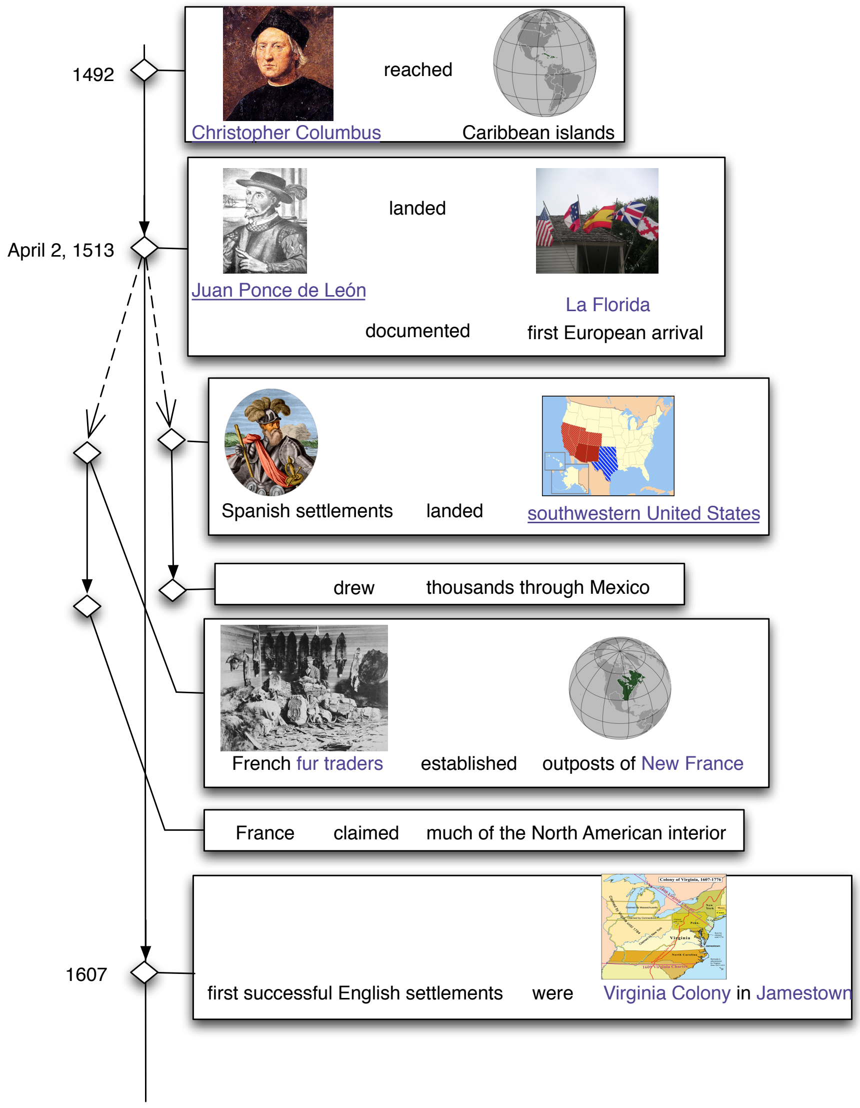

Figure 8.2: Temporal Visualization with illustration of a section of a Wikipedia article on Christopher Columbus

ing TempEval[1] (section 7.3) will provide only the raw text and will require the participants to annotate the text with temporal information, which includes extracting entities, identifying paired entities with temporal relations, and finally classifying the temporal relations.

The initial goal of TimeML (PCI$^+$03) temporal annotation scheme was to assist the Question Answering (QA) task. In the last few years, due to the complexity of the task, the focus had shifted towards solving smaller subtasks through these TempEval Shared Tasks. However, with our temporal QA toolkit (in Chapter 5), now it is prime time to improve the temporal QA systems. Future TempEval organizers can evaluate the temporal QA systems with our proposed metrics (section 6.2).

8.3 Closing Remarks

In this dissertation we presented systems to solve a portion of the language understanding problem – temporal information understanding. We extracted temporal information from raw texts, classified temporal relations between entities, and built a system to do the temporal reasoning. Additionally, we proposed evaluation methodologies to evaluate automated systems and further improved the existing temporal processing resources.

[1]`http://www.cs.york.ac.uk/semeval-2013/task1/`

Bibliography

[AAdR05] D Ahn, S.F. Adafre, and M. de Rijke. Extracting temporal information from open domain text: A comparative exploration. In *Digital Information Management*, 2005.

[All83] James F. Allen. Maintaining knowledge about temporal intervals. *Communication ACM*, 26(11):832–843, 1983.

[All84] James F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154, 1984.

[All91] James. F. Allen. Time and time again: The many ways to represent time. *Intelligent Systems*, 1991.

[AS92] R. Arthur and J.. Stillman. Temporal reasoning for planning and scheduling. 1992.

[ASB08] James Allen, Mary Swift, and Will de Beaumont. Deep semantic analysis of text. In *Symposium on Semantics in Systems for Text Processing (STEP)*, 2008.

[BA05] B. Boguraev and R. K. Ando. Timebank-driven timeml analysis. in annotating, extracting and reasoning about time and events. In *Dagstuhl Seminar Proceedings*, 2005.

[BDLB06] Philip Bramsen, Pawan Deshpande, Yoong Keok Lee, and Regina Barzilay. Inducing temporal graphs. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), Sydney.* Association for Computational Linguistics, 2006.

[BLB$^+$01] Eric Brill, Jimmy Lin, Michele Banko, Susan Dumais, and Andrew Ng. Data-intensive question answering. In *Proceedings of the Tenth Text REtrieval Conference (TREC)*, 2001.

[BM06] S Bethard and J. Martin. Identification of event mentions and their semantic class. In *In Empirical Methods in Natural Language Processing (EMNLP)*, 2006.

[BM07] Steven Bethard and James H. Martin. CU-TMP: Temporal Relation Classification Using Syntactic and Semantic Features. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 129–132, Prague, Czech Republic, 2007. Association for Computational Linguistics.

[BW98] Eric Brill and Jun Wu. Classifier combination for improved lexical disambiguation. In *Proceedings of the 17th international conference on Computational linguistics*, volume 1, pages 191–195, 1998.

[CAM07] Yuchang Cheng, Masayuki Asahara, and Yuji Matsumoto. NAIST.Japan: temporal relation identification using dependency parsed tree. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval)*, pages 245–248. ACL, 2007.

[CJ08] Nathanael Chambers and Dan Jurafsky. Jointly combining implicit constraints improves temporal ordering. In *EMNLP-08*, Palo Alto, CA, USA, 2008.

[CWJ07] Nathanael Chambers, Shan Wang, and Daniel Jurafsky. Classifying temporal relations between events. In *ACL '07: Proceedings of the 45th Annual Meeting on Association for Computational Linguistics*. Association of Computational Linguistics, 2007.

[DAS03] M Dzikovska, J Allen, and M Swift. Integrating linguistic and domain knowledge for spoken dialogue systems in multiple domains. In *Workshop on Knowledge and Reasoning in Practical Dialogue Systems, IJCAI, Acapulco*, 2003.

[DG10] Leon Derczynski and Robert Gaizauskas. USFD2: Annotating Temporal Expresions and TLINKs for TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 337–340, Uppsala, Sweden, 2010. Association for Computational Linguistics.

[DG11] L. Derczynski and R. Gaizauskas. An Annotation Scheme for Reichenbach's Verbal Tense Structure. In *Proceedings of the 6th Joint ACL-ISO Workshop on Interoperable Semantic Annotation*, pages 10–17, 2011.

[DM87] T. Dean and D. McDermott. Temporal data base management. *Artificial Intelligence*, 32, 1987.

[Dow86] David R. Dowty. The effects of aspectual class on the temporal structure of discourse: Semantics or pragmatics? *Linguistics and Philosophy*, 9:37–61, 1986.

[Fer04] L Ferro. Tern. temporal expression recognition and normalization. url: http: //timex2.mitre.org/tern.html. Technical report, MITRE, 2004.

[FGMW03] L Ferro, L Gerber, I Mani, and G Wilson. Tides 2003 standard for the annotation of temporal expressions. Technical report, MITRE, 2003.

[FH01] Elena Filatova and Eduard Hovy. Assigning time-stamps to event-clauses. In *Proceedings of the workshop on Temporal and spatial information processing*, pages 1–8, Morristown, NJ, USA, 2001. Association for Computational Linguistics.

[GSS93] Alfonso Gerevini, Lenhart Schubert, and Stephanie Schaeffer. Temporal reasoning in timegraph i–ii. *SIGART Bull.*, 4(3):21–25, 1993.

[GTAB10] Claire Grover, Richard Tobin, Beatrice Alex, and Kate Byrne. Edinburgh-LTG: TempEval-2 System Description. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 333–336, Uppsala, Sweden, 2010. Association for Computational Linguistics.

[Hau87] B. A. Haugh. Non-standard semantics for the method of temporal arguments. *IJCAI*, pages 449–457, 1987.

[HB05] Sanda Harabagiu and Cosmin Adrian Bejan. Question answering based on temporal inference. In *Proceedings of the AAAI-2005 Workshop on Inference for Textual Question Answering*, 2005.

[HBLL10] Eun Ha, Alok Baikadi, Carlyle Licata, and James Lester. NCSU: Modeling Temporal Relations with Markov Logic and Lexical Ontology. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 341–344, Uppsala, Sweden, 2010. Association for Computational Linguistics.

[HCD05] K Hachioglu, Y. Chen, and B. Douglas. Automatic time expression labeling for english and chinese text. In *In Proceedings of CICLing-2005*, 2005.

[HP03] Jerry Hobbs and James Pustejovsky. Annotating and reasoning about time and events. In *Proceedings of the AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*, 2003.

[HS92] Chung Hee Hwang and Lenhart K. Schubert. Tense trees as the "fine structure" of discourse. In *Proceedings of the 30th annual meeting on Association for Computational Linguistics*, pages 232–240, Morristown, NJ, USA, 1992. Association for Computational Linguistics.

[HSG07] Mark Hepple, Andrea Setzer, and Rob Gaizauskas. USFD: preliminary exploration of features and classifiers for the TempEval-2007 tasks. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval)*, pages 438–441. ACL, 2007.

[HT07] C. Hagege and X. Tannier. Xrce-t: Xip temporal module for tempeval campaign. In *4th Workshop on Semantic Evaluations (SemEval-2007), ACL 2007.*, 2007.

[JF00] C Johnson and C Fillmore. The framenet tagset for frame-semantic and syntactic coding of predicate-argument structure. In *ANLP-NAACL, Seattle, WA.*, 2000.

[Kau91] Henry Kautz. Mats (metric/allen time system) documentation. *Technical Report*, 1991.

[KEB10] Anup Kumar Kolya, Asif Ekbal, and Sivaji Bandyopadhyay. Ju_cse_temp: A first step towards evaluating events, time expressions and temporal relations. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 345–350, Uppsala, Sweden, 2010. Association for Computational Linguistics.

[KL91] H. Kautz and P. Ladkin. Integrating metric and qualitative temporal reasoning. *AAAI*, 1991.

[KM09] O. Kolomiyets and Marie-Francine Moens. Meeting tempeval-2: Shallow approach for temporal tagger. In *In Proceedings of the NAACL HLT Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, 2009.

[KM10] Oleksandr Kolomiyets and Marie-Francine Moens. Kul: Recognition and normalization of temporal expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 325–328, Uppsala, Sweden, 2010. Association for Computational Linguistics.

[Koo89] J. Koomen. The timelogic temporal reasoning system. Technical report, Computer Science Department, University of Rochester, Rochester, NY, USA, 1989.

[KR93] Hans Kamp and Uwe Reyle. *From Discourse to Logic: Introduction to Model Theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory.* Dordrecht, The Netherlands, 1993.

[LA93] Alex Lascarides and Nicholas Asher. Temporal interpretation, discourse relations and commonsense entailment. *Linguistics and Philosophy*, 16(5):437–493, 1993.

[LSNC10] Hector Llorens, Estela Saquete, and Borja Navarro-Colorado. TIPSem (English and Spanish): Evaluating CRFs and Semantic Roles in TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 284–291. ACL, 2010.

[LSNG11] Hector Llorens, Estela Saquete, Borja Navarro, and Robert Gaizauskas. Time-surfer: Time-based graphical access to document content. In *Annotating, Extracting and Reasoning about Time and Events*, volume 6611 of *LNCS - Advances in Information Retrieval*, pages 767–771. Springer, 2011.

[LUA12] Hector Llorens, Naushad UzZaman, and James Allen. Merging Temporal Annotations. In *Proceedings of the19th International Symposium on Temporal Representation and Reasoning*, 2012.

[McD82] Drew McDermott. A temporal logic for reasoning about processes and plans. *Cognitive Science*, pages 101–155, 1982.

[MCH05] Dan Moldovan, Christine Clark, and Sanda Harabagiu. Temporal context representation and reasoning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2005.

[MH82] J. McCarthy and P. Hayes. Some philosophical problems from the standpoint of ai. *Machine Intelligence*, 1982.

[MPG05] Inderjeet Mani, James Pustejovsky, and Rob Gaizauskas. *The Language of Time: A Reader*. Oxford Linguistics, 2005.

[MS88] Marc Moens and Mark Steedman. Temporal ontology and temporal reference. In *Proceedings of the Association for Computational Linguistics*, Morristown, NJ, USA, 1988. Association for Computational Linguistics.

[MS90] S Miller and Len K. Schubert. Time revisited. In *Proceedings of Computational Intelligence, 6(2)*, 1990.

[MSF07] Congmin Min, Munirathnam Srikanth, and Abraham Fowler. Lcc-te: a hybrid approach to temporal relation identification in news text. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval)*, pages 219–222. ACL, 2007.

[MVW⁺06] Inderjeet Mani, Marc Verhagen, Ben Wellner, Chong Min Lee, and James Pustejovsky. Machine learning of temporal relations. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 753–760, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[MW96] Tony McEnery and Andrew Wilson. *Corpus Linguistics*. Edinburgh University Press, 1996.

[Pas88] Rebecca Passanneau. A computational model of the semantics of tense and aspect. *Computational Linguistics 14(2)*, 1988.

[PCI⁺03] James Pustejovsky, Jos M. Castao, Robert Ingria, Roser Sauri, Robert J. Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R. Radev. TimeML: Robust Specification of Event and Temporal Expressions in Text. In Mark T. Maybury, editor, *New Directions in Question Answering*, pages 28–34. AAAI Press, 2003.

[PHS⁺03] J. Pustejovsky, P. Hanks, R. Sauri, A. See, R. Gaizauskas, A. Setzer, D. Radev, B. Sundheim, D. Day, L. Ferro, and M. Lazo. The TIMEBANK corpus. In *Proceedings of Corpus Linguistics 2003*, pages 647–656, Lancaster, March 2003.

[Pri68] A. N. Prior. Tense logic and the logic of earlier and later. *Chapter 11 of Papers on Time and Tense*, pages 156–174, 1968.

[PST07] J Poveda, M Surdeanu, and J. Turmo. A comparison of statistical and rule-induction learners for automatic tagging of time expressions in english. In *In Proceedings of the International Symposium on Temporal Representation and Reasoning*, 2007.

[Pus07] Georgiana Puscasu. Wvali: Temporal relation identification by syntactico-semantic analysis. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 484–487, Prague, Czech Republic, 2007. Association for Computational Linguistics.

[PWM02] James Pustejovsky, Janyce Wiebe, and Mark Maybury. Multiple-perspective and temporal question answering. In *Proceedings of the Proceedings of LREC workshop on question answering: Strategy and Resources*, 2002.

[RD06] Matthew Richardson and Pedro Domingos. Markov logic networks. In *Machine Learning*, 2006.

[Rei47] Hans Reichenbach. The tenses of verbs. In *Elements of Symbolic Logic*, 1947.

[RMS$^+$11] Sebastian Riedel, David McClosky, Mihai Surdeanu, Andrew McCallum, and Chris Manning. Model Combination for Event Extraction in BioNLP 2011. In *Proceedings of the NLP in Biomedicine ACL 2011 Workshop (BioNLP 2011)*, 2011.

[SAG04] Mary Swift, James F. Allen, and Daniel Gildea. Skeletons in the parser: Using a shallow parser to improve deep parsing. In *Proceedings of COLING*, 2004.

[Saq10] Estela Saquete. ID 392: TERSEO+T2T3 Transducer. A systems for Recognizing and Normalizing TIMEX3. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 317–320, Uppsala, Sweden, 2010. Association for Computational Linguistics.

[SC91] Fei Song and Robin Cohen. Tense interpretation in the context of narratives. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91)*, 1991.

[Sch89] R. Schrag. The timelogic temporal reasoning system. Technical report, Computer Science Department, University of Rochester, Rochester, NY, USA, 1989.

[Seb08] Riedel Sebastian. Improving the accuracy and efficiency of map inference for markov logic. In *In Proceedings of UAI 2008*, 2008.

[SG00] A. Setzer and R. Gaizauskas. Annotating events and temporal information in newswire texts. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, pages 1287–1294, Athens, 2000.

[SG10] Jannik Strötgen and Michael Gertz. HeidelTime: High Quality Rule-Based Extraction and Normalization of Temporal Expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 321–324, Uppsala, Sweden, 2010. Association for Computational Linguistics.

[SGH03] Andrea Setzer, Robert Gaizauskas, and Mark Hepple. Using semantic inferences for temporal annotation comparison. In *Proceedings of the Fourth International Workshop on Inference* *in Computational Semantics (ICoS-4)*, pages 185–196, Nancy, France, 2003.

[Sj03] Jonas Sjbergh. Combining POS-taggers for improved accuracy on Swedish text. In *Proceedings of the NoDaLiDa*, 2003.

[SKVP05] Roser Saurí, Robert Knippen, Marc Verhagen, and James Pustejovsky. Evita: a robust event recognizer for qa systems. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 700–707, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[SVMB$^+$09] E. Saquete, J. Luis Vicedo, P. Martnez-Barco, R. Muoz, and H. Llorens. Enhancing qa systems with complex temporal question processing capabilities. *Journal of Artificial Intelligence Research*, 2009.

[Tau83] J. Taugher. An efficient representation for time information. Master's thesis, Department of Computer Science, University of Alberta., 1983.

[TM08] Xavier Tannier and Philippe Muller. Evaluation Metrics for Automatic Temporal Annotation of Texts. In ELRA, editor, *Language Resources and Evaluation Conference (LREC)*, Marrakech, Morocco, 2008.

[UA10] Naushad UzZaman and James F. Allen. TRIPS and TRIOS system for TempEval-2: Extracting temporal information from text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 276–283, Uppsala, Sweden, 2010. Association for Computational Linguistics.

[UA11] Naushad UzZaman and James Allen. Event and temporal expression extraction from raw text: first step towards a temporally aware system. In *International Journal of Semantic Computing*, 2011.

[UBA10a] Naushad UzZaman, Jeffrey P. Bigham, and James F. Allen. Multimodal summarization for people with cognitive disabilities in reading, linguistic and verbal comprehension. In *2010 Coleman Institute Conference*, 2010.

[UBA10b] Naushad UzZaman, Jeffrey P. Bigham, and James F. Allen. Pictorial temporal structure of documents to help people who have trouble reading or understanding. In *International Workshop on Design to Read, ACM Conference on Human Factors in Computing Systems (CHI)*, 2010.

[UBA11] Naushad UzZaman, Jeffrey P. Bigham, and James F. Allen. Multimodal summarization of complex sentence. In *International Conference on Intelligent User Interfaces (IUI)*, 2011.

[ULA$^+$12] Naushad UzZaman, Hector Llorens, James Allen, Leon Derczynski, Marc Verhagen, and James Pustejovsky. TempEval-3: Evaluating Events, Time Expressions, and Temporal Relations. 2012.

[VBA$^+$95] Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. A model-theoretic coreference scoring scheme. In *MUC6 95: Proceedings of the 6th conference on Message understanding*, 1995.

[Ven67] Z. Vendler. *Verbs and Times*, chapter 4, pages 97–121. Cornell University Press, Ithaca, New York, 1967.

[Ver07] March Verhagen. Drawing TimeML relations with TBox. In Graham Katz, James Pustejovsky, and Frank Schilder, editors, *Annotating, Extracting and Reasoning about Time and Events*, volume 4795 of *Lecture Notes in Computer Science*, pages 7–28. Springer, 2007.

[VGS$^+$07] Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of 4th International Workshop on Semantic Evaluations (SemEval 2007)*, 2007.

[Vil94] L. Vila. A survey on temporal reasoning in artificial intelligence. *AI Communications*, 7(1):4, 1994.

[VK86] M. Vilain and H. Kautz. Constraint propagation algorithm for temporal reasoning. In *Proceedings of the Fifth National Conference on Artificial Intelligence (AAAI-86)*. AAAI, 1986.

[VKvB90] Marc Vilain, Henry Kautz, and Peter van Beek. *Constraint propagation algorithms for temporal reasoning: a revised report*, pages 373–381. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[VSCP10] Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. Semeval-2010 task 13: Tempeval 2. In *Proceedings of International Workshop on Semantic Evaluations (SemEval 2010)*, 2010.

[Web88] Bonnie Webber. Tense as discourse anaphor. *Computational Linguistics 14(2)*, 1988.

[YA93] Ed Yampratoom and James F. Allen. Performance of temporal reasoning systems. Technical report, Computer Science Department, University of Rochester, Rochester, NY, USA, 1993.

[YRAM09] Katsumasa Yoshikawa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. Jointly identifying temporal relations with markov logic. In *Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 2009.

A Contributions

This appendix lists and briefly describes the academic publications and the language resources that are generated in the course of this PhD.

A.1 Publications

- Naushad UzZaman and James F. Allen. Extracting Events and Temporal Expressions from Text. *Fourth IEEE International Conference on Semantic Computing (IEEE ICSC2010)*, Pittsburgh, USA, September 2010. (Chapter 4).
- Naushad UzZaman and James F. Allen. Event and Temporal Expression Extraction from Raw Text: First Step Towards a Temporally Aware System. *International Journal of Semantic Computing*, 2011. (Chapter 4).
- Naushad UzZaman and James F. Allen. TRIPS and TRIOS System for TempEval-2: Extracting Temporal Information from Text. *International Workshop on Semantic Evaluations (SemEval-2010), Association for Computational Linguistics (ACL)*, Sweden, July 2010. (Chapter 4).

- Naushad UzZaman and James F. Allen. Temporal Evaluation. *Proc. of The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (Short Paper)*, Portland, Oregon, USA, June 2011. (Chapter 6).

- Naushad UzZaman, Hector Llorens and James F. Allen. Evaluation of Temporal Information Understanding with Temporal Question Answering. *Proc. of IEEE International Conference on Semantic Computing, Italy, September 2012.?* (Chapter 5, 6).

- Naushad UzZaman and James F. Allen. TRIOS-TimeBank Corpus: Extended TimeBank corpus with help of Deep Understanding of Text. *Proc. of The Seventh International Conference on Language Resources and Evaluation (LREC)*, Malta, May 2010. (Chapter 7).

- Naushad UzZaman, Hector Llorens and James F. Allen. Merging Temporal Annotations. *Proc. of The 19th International Symposium on Temporal Representation and Reasoning.* (Chapter 7).

- Naushad UzZaman, Hector Llorens, James F. Allen, Leon Derczynski, Marc Verhagen, James Pustejovsky. 2012. TempEval-3: Evaluating Events, Time Expressions, and Temporal Relations. *arXiv:1206.5333v1.* (Chapter 7).

- Naushad UzZaman, Jeffrey P. Bigham and James F. Allen. Multimodal Summarization of Complex Sentences. *Proc. of International Conference of Intelligent User Interface (IUI)*, Palo Alto, CA, 2011. (Chapter 8).

- Naushad UzZaman, Jeffrey P. Bigham and James F. Allen. Multimodal Summarization for people with cognitive disabilities in reading, linguistic and verbal comprehension. *2010 Coleman Institute Conference*, Denver, CO, 2010. (Chapter 8).

A.2 Language Resources

1. **Temporal Expression Normalizer**
   This program takes a temporal expression as an input and returns the normalized *type* and *value* according to TimeML scheme. This is a *regular expression* based program, which had the second best performance in TempEval 2010 (temporal evaluation shared task). It is described in section 4.4, and the toolkit is available online at `http://code.google.com/p/roc-trios/downloads/detail?name=tempexp-normalizer.py`.

2. **Temporal QA Toolkit**
   Given a TimeML annotated document, this toolkit answers temporal questions by doing temporal reasoning. This toolkit can answer *yes/no, list* and *factoid* questions. It is described in Chapter 5 and available online at `http://www.cs.rochester.edu/u/naushad/temporal`.

3. **Temporal Evaluation Toolkit**
   This toolkit evaluates systems that extract temporal information from text. It uses temporal closures to reward relations that are equivalent but distinct. This metric measures the overall performance of a system with a single score, making comparison between different systems straightforward. This metric has been adopted to evaluate the participants in TempEval-3. The toolkit is described in section 6.1. It is available online at `http://code.google.com/p/roc-trios/downloads/detail?name=Temporal_Evaluation.zip`.

4. **Toolkit to Evaluate Temporal Understanding with QA**

   This toolkit evaluates the temporal information understanding with temporal question-answering. Our proposal for evaluating temporal information understanding with temporal QA is motivated by the benefits inherent in the use of QA to assess the understanding of natural language. The benefits of using temporal QA for temporal information understanding are: (i) QA represents a task-focused way of evaluation, which is natural and simple for humans; (ii) it is much easier to create temporal questions (needed to evaluate temporal information by QA) than annotating new documents with temporal information (needed to evaluate by corpus). Hence, the evaluation by QA allows to create a larger test set more easily. (iii) moreover, QA allows the evaluation of temporal understanding in other domains as well, which enables the testing of the generality of the systems. This toolkit is described in section 6.2 and available online at `http://www.cs.rochester.edu/u/naushad/temporal`.

5. **TRIOS-TimeBank corpus and TRIPS Ontology**

   TRIOS-TimeBank corpus is an extension to TimeBank corpus with suggestions for missed events and temporal expressions, new event feature (ontology type), extra SLINKs, and new RLINK. Details can be found in section 7.1. TRIPS ontology has a mapping from ontology type to WordNet. It is available online at `https://www.cs.rochester.edu/u/naushad/trios-timebank-corpus`.

6. **TempEval-3 Resources**

   Temporal Evaluation Shared Task - TempEval-3 is described in section 7.3. The released resources such as corpora and evaluation toolkit for TempEval-3 will be available online at `http://www.cs.york.ac.uk/semeval-2013/task1/`.