UNDERSTANDING TIME IN NATURAL LANGUAGE TEXT

BY

QIANG NING

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2019

Urbana, Illinois

Doctoral Committee:

Adjunct Professor Dan Roth, Chair
Professor Mark Hasegawa-Johnson
Associate Professor Julia Hockenmaier
Professor Wen-mei Hwu
Professor Martha Palmer, University of Colorado Boulder

ABSTRACT

Understanding *time* is essential to understanding events in the world. Knowing what has happened, what is happening, and what may happen in the future is critical for reasoning about those events. It is thus an important natural language processing (NLP) task to understand time.

This thesis advances the study of time by developing new insights into some aspects of the problem of reasoning about time in text, new algorithmic and machine learning approaches, and new datasets that would support continuing work on these problems by the research community. We also discuss a few research directions suggested by this work that could further improve our understanding of time in natural language text.

The thesis specifically addresses three key aspects of the temporal reasoning problem: time expression understanding, temporal relation extraction, and temporal common sense acquisition. Time expressions (e.g., *yesterday* or *last month*) often provide absolute time anchors for events. Temporal relations (e.g., event A is *before* or *after* event B) provide relative order information between events, which is complementary to time expressions. Temporal common sense (e.g., duration and frequency) is another important component in temporal reasoning, but is usually absent in a single piece of text because people do not say things that are obvious. The bulk of this thesis is devoted to the important problem of identifying temporal relations between events, a problem that has been studied a lot by the research community. The work in the thesis introduces new machine learning methods and a novel conceptual view of the problem that together result in an improvement of more than 20% over the previous state-of-the-art.

To my family, for their love and support.

ACKNOWLEDGMENTS

Firstly, I would like to express my sincere gratitude to my advisor Prof. Dan Roth for his patience, motivation, and immense knowledge. His support helped me out in the hardest times of my Ph.D. study. In my three years in Dan's group, Dan guided me to do research, to collaborate, and to mentor younger students. This is my most enjoyable experience in Urbana (even more enjoyable than the snow here). I wish I could have continued these good memories, but like every Ph.D. student, I have to start a new chapter with my family, so now, I am trying to conclude my graduate study with this thesis.

Besides my advisor, I would like to thank the rest of my doctoral committee: Prof. Mark Hasegawa-Johnson, Prof. Julia Hockenmaier, Prof. Wen-mei Hwu, and Prof. Martha Palmer. It has been a great honor to have them on my committee, and their insightful comments and encouragement have greatly helped my thesis work. Prof. Hockenmaier and Prof. Hwu have also kindly offered invaluable guidance through my job search, from which I have benefited a lot.

I would like to express my gratitude to my collaborators, Prof. Nanyun Peng, Dr. Haoruo Peng, Dr. Daniel Khashabi, Dr. Prithwish Basu, Dr. Eric Graves, Dr. Jinjun Xiong, Dr. Omer Anjum, Hao Wu, Hangfeng He, Ben Zhou, Zhili Feng, Sanjay Subramanian, Rujun Han, and Mark Yu, for the stimulating discussions and for the sleepless nights we were working together before deadlines. Also I thank my fellow labmates, Dr. Mark Sammons, Dr. Snigdha Chaturvedi, Dr. Wenpeng Yin, Dr. Stephen Mayhew, Dr. Shyam Upadhyay, Soham Dan, Daniel Deutsch, Xiaodong Yu, and Sihao Chen, for the generous help they provided me in various ways over the past three years.

I am very grateful to have the constant support of my parents and my brother. I also always feel so fortunate that my wife and I have been able to pursue our Ph.D. degrees together in Urbana. Her devotion to our family and her faith in me have been priceless, without which I would be incomplete.

Finally, thanks to the Department of ECE and the Department of CS at the University of Illinois at Urbana-Champaign, for their second-to-none academic resources, and to the IBM-Illinois C3SR center, the Army Research Lab, and the Defense Advanced Research Projects Agency (DARPA), for funding my research at different stages.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation

Time is an important dimension when we describe the world and has been involved ubiquitously in many techniques. For instance, a classical framework in signal processing is Fourier transform: $X(f) \triangleq \int_{-\infty}^{\infty} x(t)e^{-2\pi ft}dt$, where $x(t)$ is transformed from the time-domain to $X(f)$ in the frequency-domain. We can see the indispensable role of time "t" as we need values of $x(t)$ at all time points in the equation above, which is usually achieved by knowing the closed-form expression of $x(t)$. In causality analysis, a time-series $\{x_n\}_{n=0}^N$ is called Granger causal for another time-series $\{y_n\}_{n=0}^N$ if $\{x_n\}_{n=0}^N$ helps predict $\{y_n\}_{n=0}^N$ at some stage in the future via linear regression models; again, time is important because the time-series are the values sampled uniformly from the trajectories of some random processes.

In practice, there are (probably) no closed-form representations of all the events happening in the world. Instead, natural language is used to describe and reason about the world, for instance, in news articles, social media, financial reports, and electronic health records. We probably do not have to be worried about things like Fourier transform anymore, but time is still a crucial dimension to think about. For instance, *"he won the championship yesterday"* is very different from *"he will win the championship tomorrow"* in the sense that the first happened in the past and cannot be changed (if we trust the speaker's words), while the second will be in the future and is not guaranteed to be achieved (even if we trust the speaker's words because we understand that the speaker is simply making a prediction). When multiple events are

involved in a story, the temporal order among them also matters, altering which may result in a different understanding of the same set of events. For instance, let $A \rightarrow B$ denote that A is temporally before B. If *"people were angry"*$\rightarrow$*"police suppressed people"*, then it leaves the readers an impression that people got angry and perhaps ended up in a violent confrontation with the police, and then the police wanted to restore order by suppressing them. Instead, if *"police suppressed people"*$\rightarrow$*"people were angry"*, then it means that people got angry because of the suppression. The tone of the story and the side that should be blamed are different simply because of a different temporal order. Similarly, *"the doctor found a tumor"*$\rightarrow$*"he had a surgery"* tells us that the purpose of the surgery was to remove the tumor (so now there is probably no tumor anymore), while *"he had a surgery"*$\rightarrow$*"the doctor found a tumor"* sounds like a tumor was found unexpectedly during the surgery (so there is probably still a tumor now). All these examples indicate the important role that time is playing in natural language processing (NLP).

When people use natural language to describe and reason about the world, it is assumed that, in most cases, human readers are able to understand these temporal issues. However, how to make computers understand semantics related to time has not received enough attention in the NLP community and remains an open challenge. This leads to the problem statement of this thesis, as we detail below.

1.2 Problem Statement

With ever-growing natural language data available nowadays in the form of news articles, books, online reviews, and social network posts, it is increasingly important to make use of these data to understand how things are evolving and, hopefully, to make decisions based on it. This thesis argues that a key issue here that has been less studied is to understand the temporal information in these natural language data, and the core question of it is to know *when something happens*.

To answer this "when" question requires two very basic components: time expression (TIMEX) understanding and temporal relation (TEMPREL) understanding

[1, 2, 3]. The first, also known as the TIMEX component, requires understanding those explicit time expressions so that these TIMEX-es can serve as the timestamps that we are expecting. In Example 1, *t1:February 27, 1998* is such a TIMEX. Note that in order to allow computers to understand these TIMEX-es, a conversion to a standard format is also needed besides simply chunking them out from text, e.g., *February 27, 1998* needs to be converted to something like "1998-02-27", and this step is called TIMEX *normalization*; in contrast, identifying the span of *February 27, 1998* in the original text is called TIMEX *extraction*. TIMEX extraction and normalization are the two steps in the TIMEX component.

Example 1:
A car (*e1:exploded*) in the middle of a group of men playing volleyball on (*t1:February 27, 1998*) and more than 10 people have (*e2:died*).

The second basic component to answer the "when" question is the TEMPREL component, which conceptually aims at determining which event or action happens earlier in time. While TIMEX-es often act as absolute time anchors which carry temporal information *explicitly*, TEMPRELs provide another type of *implicit* temporal information, i.e., the relative order of events, which is important especially in absence of TIMEX-es. That is, given a pair of events or actions, determine which of them occurs first (or other temporal relations between them, e.g., simultaneous or overlapping). In Example 1, there are two events: *e1:exploded* and *e2:died*. The text tells us that *e1* was on *1998-02-27* but does not tell us when *e2* happened exactly. In this case, how do we know when *e2* happened? It turns out that human readers do not usually feel ambiguity here because we know that there is a TEMPREL between them, i.e., *e1:exploded* happened *before e2:died*. The TIMEX component and the TEMPREL component together provide a more complete picture of the temporal aspect of a story, either explicitly or implicitly, so they are naturally the most important building blocks for understanding time.

In addition to discussing the two components above, this thesis also notes the necessity of commonsense knowledge for time, or in other words, *temporal common sense*. For instance, the temporal order between some events can often be determined

purely based on the event verbs, and this is due to the temporal order common sense (e.g., *death* should be *after* (instead of *before*) *explosion*). Another example of temporal common sense is duration, i.e., how long something lasts. Because we rarely say things that are common sense to others, it is often difficult for a computer to fill in the blanks when trying to understand such cases.

1.3 Challenges

A key difference in terms of the time dimension between NLU and other techniques is the availability of gold timestamps: In techniques such as signal processing and time-series analysis, gold timestamps often naturally come along with the data, while in natural language text, gold timestamps often do not exist. Next we will specifically discuss the various challenges in each component of this thesis.

Some types of text, for example tweets or news articles, can come with a machine-readable timestamp which tells the readers when this tweet or news article was published. These publication timestamps are of course important, but they are not accurate enough, because a tweet published on day X is very likely to talk about things that happen on day Y. Unfortunately, people often do not include the timestamps explicitly in natural language text, which leaves to readers the problem of figuring out when something happens. For instance, it does not sound natural if someone says *"people were angry at 2013-01-02T08:00:00; the police suppressed people at 2013-01-02T08:05:00"*. Instead, it is more natural if one says *"people were angry and then the police suppressed people on January 2nd, 2013"*, where the speaker neither reveals the exact timestamp for each event, nor follows the ISO format for time. Therefore, it is left to the computer to figure out which part of the sentence is a TIMEX, which in this case is *January 2nd, 2013*.

Assuming a computer knows *January 2nd, 2013* is the TIMEX, then to make full use of it, the computer still needs to convert it into a machine-readable timestamp such as *2013-01-02*. This step is called TIMEX normalization, which is usually not so trivial as it seems to be in *January 2nd, 2013*. For instance, if the TIMEX was

January 2nd or *Wednesday*, we would need to figure out the reference time, and in this particular case, the year or the week of it; similar cases are "3 days ago", "next June", or even "the Christmas before election." Therefore, TIMEX normalization is another challenge for TIMEX understanding.

In addition to the challenge in understanding TIMEX, the TEMPREL task is even more difficult. Even the top systems only achieved F_1 scores of around 35% in the TempEval3 workshop [3] which focused on the news domain. The difficulty of it is two-fold. First, we do not always describe events sequentially in their temporal order; instead, we may describe things that happened later first and then those that happened earlier. What makes things worse is when we alter the narrative order of events, we do not always add an explicit connective (e.g., *before, after, or during*) between them. As a result, TEMPRELs often have to be inferred, from lexical cues, between the lines, and sometimes even purely based on background knowledge. In Example 2, knowing that people usually become friends before getting married, we understand that *e3* is before *e4*. For a computer, however, identifying this TEMPREL is very difficult, since it is unclear when "they were in college" and there are no syntactic cues indicating the order; let alone the fact that the narrative order of the two events can be reversed without changing the meaning of the text (as shown by *e5* and *e6*). How to acquire and inject human prior knowledge into TEMPREL extraction is thus the first challenge.

Example 2:
They (*e3:became*) friends in college. They got (*e4:married*) in 2015.
They got (*e5:married*) in 2015. They (*e6:became*) friends in college.

Second, collecting enough high-quality TEMPREL annotations is also very challenging. On one hand, annotating the TEMPRELs among n events requires $\mathcal{O}(n^2)$ individual annotations, which makes it difficult to scale data annotation up to large datasets. Although existing datasets have tried to annotate only those events that are close-by in text, the annotation task remains so time consuming that existing datasets are all relatively small. On the other hand, for each individual TEMPREL, the annotation not only requires a TEMPREL label (e.g., *before* or *after*), but also in-

volves decisions like what the events are and whether the TempRel actually exists. For instance, in *"I wanted to leave this place"*, should we consider *leave* as an event? In *"the police wanted to eliminate the anti-government army but failed"*, is there a TempRel between *eliminate* and *failed*? If not, what determines the existence of a TempRel? These complications have resulted in low inter-annotation agreement (IAA) in existing datasets, hence another challenge for TempRel understanding.

Finally, the knowledge of human common sense has always been considered as an important missing piece for computers, and it is also the case for temporal common sense. Besides temporal order common sense demonstrated in Example 2, there are other types of temporal common sense such as duration and frequency that are also missing for existing NLP techniques. Below is an example for duration common sense: a human can easily tell that the first blank should be "will not" while the second blank should be "will" because a human knows *a break* is usually short while *a vacation* may last days or even weeks. The difficulty of acquiring temporal common sense is that this knowledge is not composed by facts (which can be handled by a look-up table); instead, it is often coarse-grained and fuzzy. For example, we know *a break* is relatively short but it could still range from a few seconds to a few hours; if we say *"Dr. Porter is now taking a Christmas break"*, then this *break* can also last a few days. There has been a lack of supervision signals that are available to guide machine learning systems to handle these ambiguities.

Example 3: choosing from *"will" or "will not"*
Dr. Porter is now (*e7:taking*) a vacation and _____ be able to see you soon.
Dr. Porter is now (*e8:taking*) a break outside and _____ be able to see you soon.

1.4 Thesis Statement

As we are clear about the problems to be addressed in this thesis and their potential challenges, our thesis statement is as follows. Understanding time from natural

language text is an important task and it requires us to understand time expressions, temporal relations, and temporal common sense. The rich structure of time provides us with temporal cues from unlabeled and noisy data, exploiting which can greatly benefit the task. To successfully exploit the rich structure of time in a learning-based approach, we should also develop both theoretical and algorithmic understandings of incidental supervision.

1.5 Outline

The rest of the thesis is organized as follows. Chapter 2 will provide more background information of this thesis, specifically on two topics: event understanding (Sec. 2.1) and structured machine learning (Sec. 2.2).

Chapter 3 describes our approach to TIMEX understanding. While the TIMEX task has been handled well by state-of-the-art systems [4, 5, 6, 7, 8] with end-to-end F_1 scores around 80%, the *CogCompTime* system proposed in this thesis work achieves performance comparable to that of state-of-the-art TIMEX systems, but is almost twice as fast as the second fastest, HeidelTime [4].

Chapter 4 addresses the aforementioned difficulties of the TEMPREL task, and our contribution is from three aspects. First, *machine learning* (Sec. 4.2). TEMPRELs are inherently structured, i.e., one TEMPREL may be affected by other TEMPRELs. We propose to better exploit the structure induced by the transitivity property of temporal relations in the structured learning framework [9, 10]. Second, *injection of commonsense knowledge* (Sec. 4.3). We collect a probabilistic knowledge base called TEMPROB to encode humans' prior knowledge of the typical ordering of events, which has also proved to be a useful resource for TEMPREL extraction [11]. Third, *data annotation* (Sec. 4.4). We investigate existing TEMPREL datasets and propose a multi-axis modeling for the temporal structure of stories [12]. Integrating these components, our current system significantly improves the state-of-the-art temporal relation extraction performance by more than 20% in F_1 (Sec. 4.5).

In terms of temporal common sense, we systematically study the problem in Chap-

ter 5 and summarize five types of temporal common sense. We develop a new dataset dedicated for these phenomena via crowdsourcing, with guidelines designed meticulously to guarantee its quality. Using this new dataset as a testbed, we find that the best existing techniques are still far behind human performance on temporal common sense understanding, indicating the need for further research in order to improve the currently limited capability to capture temporal semantics.

Finally, while studying how to understand time in natural language, we devote Chapter 6 to an important topic in machine learning: learning from indirect supervision signals. From an information theoretic point of view, we show that learning from partially annotated structural data has its unique advantage over learning from fully annotated structures, and this advantage is proved empirically on several different NLP tasks including TempRel extraction. Chapter 7 concludes this thesis and points out to some directions for future research.

CHAPTER 2

BACKGROUND

One NLP task that is closely related to understanding time in natural language is *event understanding*. The reason is two-fold. First, a time point itself is often not very meaningful until it is associated with something that happens. For instance, *February 27, 1998* itself is an ordinary time point, but *"a car exploded on Februrary 27, 1998"* makes this time point special because of the car explosion. This is why event understanding is important for the TIMEX component. Second, event understanding is also important for the TEMPREL component because every TEMPREL is between two events. Given the close relation between the topic of this thesis and event understanding, we devote this first section of this chapter to a discussion about those event understanding works in the literature, and we will see how this thesis work fits in the more general topic of event understanding.

In terms of the methodology to study time in natural language, we find that time has a very rich structure, exploiting which can significantly help us in learning and inference. Therefore, the rest of this chapter aims at providing more background information about the role of structure in machine learning.

2.1 Event Understanding

Event understanding has long been an active area in NLP and information retrieval. Generally speaking, an event is defined as an action associated with corresponding participants involved in this action. Its core question is to understand *what is going on*, which involves elements such as agents, patients, actions, location and time.

Typical tasks on event understanding are event extraction and relation extraction, which we will describe below.

2.1.1 Event Extraction

Due to the complexity and fundamental difficulties, most existing research in this area focuses on a limited domain of events. For example, in both ACE 2005 [13] and TAC KBP [14], only 8 types/33 subtypes of events are considered: Life, Movement, Business, Conflict, Personnel, Transaction, and Justice. The rich Entities, Relations and Events (rich ERE) annotation task adds an extra type of Manufacture [15]. Example 4 shows a real MOVEMENT event annotation in the ACE 2005 dataset.

Example 4: An event in ACE05/CNN_LE_20030504.1200.01
Type: MOVEMENT. Subtype: TRANSPORT.
In the case of 1991, the task was to go in and get them out of Kuwait, and they did it, and [PERSON: they] were properly greeted [ANCHOR: coming back] to [DESTINATION: the United States].

Under this definition, event extraction is usually treated as detecting event triggers (e.g., *coming back*) and determining event types (e.g., MOVEMENT) and arguments (e.g., *they* and *the United States*). In general, event trigger and argument detection is a text span detection problem, and event type detection is a multi-class classification problem. The majority of the early works take a pipelined approach where triggers are identified first and then arguments [16, 17, 18, 19, 20], and [21, 22] later propose to extract both triggers and arguments jointly. Finally in recent years, neural approaches become more and more popular on this task as well [23, 24, 25, 26, 27, 28, 29, 30].

Another line of event extraction work is via semantic role labeling (SRL) [31, 32, 33]. SRL is to represent the semantic meanings of language and answer questions like *Who did What to Whom* and *When, Where, How* [34]. In Fig. 2.1, we show an example SRL output from the Illinois Curator package[1] [35, 36]. Two semantic

[1] `http://cogcomp.org/page/demo_view/SRL`

frames are extracted: *decline.02* and *comment.01*, where the number following each verb is the (disambiguated) sense number of that verb. For example, the second sense of *decline* has two arguments, argument 0 (or Arg0/A0 in short) is *entity turning down* and Arg1 is *thing turned down* (details of these disambiguated semantic frames can be found in PropBank [37]), from which we can clearly see the natural connection between semantic frames and events. Depending on the events of interest, the SRL results are often a superset of those events of interest under the conventional definition and thus need to be filtered afterwards [33]. However, perhaps a more important question is how to understand events that are more general and not only in the very few predefined types. Existing work on *open information extraction* (OpenIE) can be seen as along this line, e.g., [38, 39, 40, 41, 42].

Figure 2.1: The semantic role labeling result of the sentence *"A Dow spokeswoman declined to comment on the estimates"* obtained via the Illinois Curator package [35, 36]. Since a semantic frame generally describes an action and the participants involved, it is naturally an event (or an event candidate when people are only interested in events of specific types).

2.1.2 Event Relations

In addition to event extraction, it is important to understand the relationship among events, e.g., co-reference resolution, causality [43, 44, 45, 46, 47, 48, 49], and event sequence modeling [50, 51, 52, 53, 54, 55]. One of the topics of this thesis is temporal relation extraction. Since temporal relations can be seen as a special type of binary event relations (i.e., a relation between two arguments), here we will describe another binary relation, event co-reference, to provide some background of event relations.

Example 5: Entity co-references are bold-faced.
I voted for **Nader** because **he** was most aligned with my values.

Co-reference in NLP is the problem of identifying whether two mentions in text are the same, in which case we say the latter one is referring to the earlier one. The NLP community have so far focused on entity co-reference (Example 5) and event co-reference (Example 6). There are multiple scopes for co-reference: sentence-level, document-level, and multiple-document-level. A bigger scope creates more difficulty in the task. Since an event involves participants and these participants are usually entities, event co-reference usually requires entity co-reference decisions and is thus more difficult. Early works on event co-reference started from scenario-specific events [56, 57] and from the sentence-level [58, 59] and afterwards have progressed to more general scenarios and the document-level. As a binary relation, co-reference is mathematically an equivalent relation, which is reflexive, symmetric and transitive. As a result, co-reference relation provides a partition of the set of events into disjoint clusters, and clustering methods can be adopted [60, 61]. There have also been works that try to exploit other structural information about events and solve co-reference with other problems jointly, for example, solving entity and event co-reference jointly [62, 63] and solving event extraction and co-reference jointly [64]. Event co-reference at the multiple-document-level has been explored (e.g., [65, 66, 67]) but still remains very challenging and far from solved.

Example 6: Event co-references are bold-faced.
The FAA on Friday **announced** it will close 149 regional airport control towers. The FAA had been expected to announce the closure of 189 low- or moderate-volume towers, but before Friday's **announcement**, it said it would consider keeping a tower open if the airport convinces the agency it is in the "national interest" to do so.

Event co-reference is closely related to the TempRel component of this thesis. First, if two events are co-references of each other, it means they are the same, including their temporal aspect, so their temporal relation must be *simultaneous*. Second, direct temporal relations are often explicitly provided within a local context (via syntactic parse structures or discourse relations), so when two events are far away in text, it is often difficult to infer their relations; event co-reference can provide long-distance relations among events and bridge events that are distant. Third, as

we introduced above, the structure induced by co-reference relations has led to much progress in terms of clustering-based methods and joint methods. We can see in Chapter 4 that our contributions in TempRel extraction can be seen as another example of exploiting the structure induced by temporal relations.

2.2 Structured Machine Learning

2.2.1 What Is Structure?

Conventionally in machine learning, one needs to make a prediction y, either discrete or continuous, based on an input x. In practice, however, one often faces the problem where the prediction y is complex, in the sense that it has multiple interrelated components that altogether constitute y. This type of machine learning problem is called structured prediction. Burton-Roberts (2016) uses an example of a bicycle to explain what a structure is [68]. In his words, a structure is divisible into parts and these parts are arranged in a certain way.

> "*Suppose you gathered together all the components of a bicycle: metal tubes, hubs, spokes, chain, cable, and so on. Try to imagine the range of objects you could construct by fixing these components together. Some of these objects might be bicycles, but others wouldn't remotely resemble a bicycle . . . only some of the possible ways of fitting bicycle components together produce a bicycle. A bicycle consists . . . in the structure that results from fitting them together in a particular way.*"
>
> — Burton-Roberts (2016)

Example 7 shows the part-of-speech (POS) tagging result of a sentence. The structure here is the entire sequence of POS tags, which can obviously be decomposed into the POS tag for each single token. In addition, these POS tags must be arranged in a certain way because not all of them are valid POS sequences in natural language (e.g., we cannot have a long sentence with all nouns). This leads to the following formal definition of structure.

Example 7: Part-of-speech tagging.					
A	car	exploded	on	Friday	.
(DT	NN	VBD	IN	NNP	.)

Definition 1 *A structure of size d is a vector $\mathbf{y} = [y_1, \ldots, y_d] \in C(\mathcal{L}^d)$, where $\mathcal{L} = \{\ell_1, \ldots, \ell_{|\mathcal{L}|}\}$ is the label set for each variable and $C(\mathcal{L}^d) \subseteq \mathcal{L}^d$ represents the constraints imposed by this type of structure. Any d-dimensional vector that does not belong to $C(\mathcal{L}^d)$ is an invalid structure.*

2.2.2 Why Is Structure Useful?

If the knowledge that $\mathbf{y} \in C(\mathcal{L}^d)$ is correct, then it is generally useful to consider this constraint in structured prediction. A naive example to show the benefit is as follows. Assume we have two variables that we want to predict, $y^{(1)}, y^{(2)} \in \mathbb{R}$, which are contaminated by two independent and identically distributed and zero-mean white Gaussian noises. Let $\mathbf{x}^{(i)} \in \mathbb{R}^n$ be the n observations for $y^{(i)}$, $i = 1, 2$, stacked as a vector. If $\mathbf{y} = [y^{(1)}, y^{(2)}]$ is not structured (or we do not use the structure even if there exists one), then the minimum mean squared error estimate for $\mathbf{y}$ is:

$$\hat{y}^{(i)} = \frac{1}{n} \sum_{j=1}^{n} x_j^{(i)}, \ i = 1, 2.$$

That is, we simply take the mean of all observations for each y. As a result, the variance of this estimator is $\mathrm{Var}(\hat{y}^{(1)}) = \mathrm{Var}(\hat{y}^{(2)}) = \sigma^2/n$, where σ is the standard deviation of the contaminating Gaussian noise. However, if we know beforehand that $\mathbf{y}$ is structured in the sense that $\mathbf{y} \in C(\mathbb{R}^2) \triangleq \{(y_1, y_2) \in \mathbb{R}^2 | y_1 = y_2\}$, then the best estimate for $\mathbf{y}$ is

$$\hat{y}^{(1)} = \hat{y}^{(2)} = \frac{1}{2n} \sum_{j=1}^{n} (x_j^{(1)} + x_j^{(2)}),$$

and the variance is now $\mathrm{Var}(\hat{y}^{(i)}) = \sigma^2/2n$. From this we can see that by incorporating knowledge about the structure, we are able to achieve a better prediction for $\mathbf{y}$ in

the sense that the estimation variance is smaller. In this example we assume $y_1 = y_2$, but generally, if the structure dictates that $\mathbf{y}$ lies in a subspace, this argument still holds given the theory of constrained Cramer-Rao bound for parameter estimation [69].

2.2.3 Inference with Structure

Assume the score of a structured prediction is parametrized by a weight vector $\mathbf{w}$ and a feature vector $\phi(\mathbf{x}, \mathbf{y})$ as $S(\mathbf{y}; \mathbf{x}, \mathbf{w}) = \mathbf{w}^T \phi(\mathbf{x}, \mathbf{y})$. Then the inference problem is to choose the best structure $\hat{\mathbf{y}}$ that maximizes $S(\mathbf{y}; \mathbf{x}, \mathbf{w})$ subject to the structural constraints. That is,

$$\hat{\mathbf{y}} = \arg \max_{\mathbf{y} \in C(\mathcal{L}^d)} \mathbf{w}^T \phi(\mathbf{x}, \mathbf{y}). \tag{2.1}$$

Due to the large space of the structure and the complication of constraints $C(\mathcal{L}^d$, this optimization problem generally cannot be solved both exactly and efficiently. If the output follows some special structure (e.g., linear chain structures), then the problem can sometimes be solved exactly using dynamic programming (e.g., the Viterbi algorithm). To solve it efficiently, one has to resort to approximate inference algorithms, such as Lagrangian relaxation, dual decomposition, or belief propagation [70, 71, 72, 73]. To solve it exactly, one popular approach is to convert Eq. (2.1) into an integer linear programming (ILP) problem and solve the resulting ILP instead [74]. ILP is in general an NP hard problem, but there exist off-the-shelf software packages that can solve ILPs in a reasonable time for small-scale problems. For example, Gurobi [75] first ignores integrality and then uses a branch and bound algorithm and cutting planes to tighten the relaxations [76]. There also exists work that can solve ILPs more efficiently by making use of previously solved ILPs [77, 78, 79].

2.2.4 Learning with Structure

If one has reason to believe that the structural constraint is correct, then enforcing the constraint in Eq. (2.1) can only be beneficial: if the unconstrained solution does not

violate this constraint, then the solution still holds in the constrained optimization problem; if the unconstrained solution violates the constraint, then it must be wrong and the constraint that we enforce explicitly may potentially correct this mistake. However, it remains unclear in theory whether enforcing the constraints in learning is also beneficial.

Therefore, one approach to structured learning is to decouple learning from inference. That is, the weight vector $\mathbf{w}$ is considered as weights for local classifiers that ignore the structured constraint in the learning phase. Once $\mathbf{w}$ is learned, the structure constraint is applied only in inference. This approach is called the learning plus inference (L+I) [80]. Another approach to structured learning is to use constraints to provide feedback to the learning protocol in an iterative inference procedure, which is termed as inference based learning (IBT) [80]. Punyakanok et al. (2005) [80] show that when the local classifier is hard to learn and the amount of structured data is sufficiently large, IBT outperforms L+I.

CHAPTER 3

TIME EXPRESSION UNDERSTANDING

As we introduced in Chapter 1, the first topic of this thesis is how to understand time expressions (TIMEX-es) in natural language. TIMEX understanding is typically decomposed into two steps. First, identify the spans of text that correspond to TIMEX-es (i.e., TIMEX extraction); second, convert the TIMEX-es in natural language into a machine-readable format (i.e., TIMEX normalization). This chapter describes existing work and how we have implemented these two steps. A highlight of this work is that our system here achieves performance comparable to that of the state-of-the-art, while using much less time. This system is later incorporated in the CogCompTime software, which Section 4.5 will explain in detail.

3.1 Existing Work

According to the approach taken, existing work can be categorized as rule-based and learning-based. Rule-based systems use regular expressions (regex) to detect trigger words such as "day" and "week", use deterministic rules to expand the trigger words into complete TIMEX phrases, and then normalize them to a machine-readable format. A typical rule-based system for this task is HeidelTime [4]. First, four types of TIMEX-es—Date, Duration, Time, and Set—will be extracted by regex (note the regex may use POS tagging from a preprocessing step). Then, for TIMEX-es like *next June*, HeidelTime chooses the reference times for each of them to be either the document creation time (DCT) or the previously mentioned DATE TIMEX, again using pre-defined rules. Finally, HeidelTime removes those phrases that are

considered invalid. For instance, if both *in March* and *March* are included, then one of them will be removed depending on the annotation guidelines. SUTime [5] is very similar to HeidelTime, but one difference is that after regex extraction, SUTime filters out extracted phrases that are unlikely to be temporal. For instance, if the POS tag for *fall* is not a noun, then it must not be a TIMEX.

Learning-based systems use classification models to chunk out TIMEX-es in text and normalize them based on grammar parsing [81, 8]. For example, UWTime [8] uses a context-dependent semantic parser for both extraction and normalization. To construct a set of meaning representations, they make use of a hand-engineered Combinatory Categorial Grammar (CCG) with about 300 hand-crafted entries in its lexicon, along with auto-generated ones such as numbers and common date formats. For example, their CCG grammar maps the phrase *2nd Friday of July* to the meaning representation INTERSECT(N-TH(2, FRIDAY), JULY). In terms of TIMEX extraction, UWTime trains a logistic regression model to detect all phrases that can be parsed by its CCG grammar. There has also been more progress in learning-based approaches in the most recent years. The Parsing Time Normalizations shared task in 2018 [82] proposed a new approach to time normalization based on the Semantically Compositional Annotation of Time Expressions (SCATE) schema [83], in which times are annotated as compositional time entities. Laparra et al. (2018) [84] proposed the first model trained on SCATE, using character-level RNNs. Xu et al. (2019) [85] pushed the idea even further by incorporating contextualized character embeddings and achieved significant improvement in TIMEX normalization.

3.2 Extraction and Normalization

UWTime provides a good balance between hand engineering and learning: the lexicon and operations in its grammar are hand-engineered and encode expert domain knowledge, based on which the extraction and normalization modules are both learned in a data-driven fashion. This thesis adopts a mixed strategy: we use machine learning in TIMEX extraction and hand engineering in TIMEX normalization. This

mixed strategy, while maintaining performance comparable to that of the state-of-the-art, significantly improves the computational efficiency of the TIMEX component, as we show later.

Technically, the TIMEX extraction step can be formulated as a generic text chunking problem and the standard B(egin), I(nside), and O(utside) labeling scheme can be used (see Example 8). This thesis proposes TemporalChunker, by retraining Illinois-Chunker [86] on top of the TIMEX chunk annotations provided in the TempEval3 workshop [3]. Let $s = [s_1, s_2, \ldots, s_n]$ be a sentence with n tokens and let $y = [y_1, y_2, \ldots, y_n]$ be the sequence of B/I/O labels for each token in s. The task of TIMEX extraction is to learn a mapping from s to y. This is a structured prediction problem because not all sequences of B/I/O labels are valid. The obvious constraint here is straightforward: O cannot be followed by I at any time. To cope with this constraint, the features used in TemporalChunker, in addition to those extracted from s, also include the B/I/O predictions from previous tokens. Since the constraint is relatively weak in this problem (we will discuss the strength of a structure in Chapter 6), inference can be performed sequentially for each token in practice without actually violating the constraint, making structural inference methods such as the Viterbi decoding unnecessary for TemporalChunker.

Example 8: The B/I/O scheme in the proposed TemporalChunker.								
A	car	exploded	on	Feb	27	,	1998	.
O	O	O	O	B	I	I	I	O

The benefit of using a learning-based chunker is in its computational efficiency. In regex matching, one has to check every substring of text against regular expressions and is often slow. As for CCG parsing, [87] presented the first polynomial-time CCG parsing algorithm. The runtime complexity of this algorithm is in $O(n^6)$, where n is the length of the input sentence. However, [88] proved that if the size of the grammar is taken into account, then any parsing algorithm for CCG will take in the worst case exponential time. Therefore, the proposed TemporalChunker can significantly improve the computational efficiency of TIMEX extraction. However, learning-based extraction handles corner cases less well than rule-based systems because of the limited training examples, which is a drawback of the proposed extraction method.

After TIMEX-es are extracted, we apply rules to normalize them. Rules are more natural for normalization: On one hand, the desired formats of various types of TIMEX-es are already defined as rules by corresponding annotation guidelines; on the other hand, the intermediate steps of how one TIMEX is normalized are not annotated in any existing datasets (it is inherently hard to do so), so learning-based methods usually have to introduce latent variables and need more training instances as a result. Therefore, this thesis adopts a rule-based normalization method. However, an obvious drawback of the proposed normalization method is that the rule set needs to be redesigned for every single language.

Specifically, we apply four types of TIMEX normalization rules for DATE, TIME, SET, and DURATION, which we will describe non-exhaustively next.

For DATE TIMEX-es, the most straightforward ones are in the canonical format such as *yyyy-MM-dd* or *MM/dd/yyyy* which are very easy to normalize. If the phrase contains special date words, *today*, *tomorrow*, or *yesterday*, we will add or subtract 1 (or 0) day from the DCT as their normalized values. For TIMEX-es like *fourth day of this week*, we feed *this week* to other date rules, and add 4 days to it. Next, regular expressions are used to detect weekdays, months, special times like *morning* and *summer*, and normalize them to corresponding formats. If words like *previous*, *last*, or *following* are detected, we also add or subtract 1 unit of time.

For TIMEX-es that represent a time point of a day (e.g., *8 am*), we use a set of regex to catch numbers connected by ":", or numbers followed by *pm*, *am*, *p.m.*, or *a.m.*. In addition, we also incorporate the JodaTime DateTimeZone package to detect timezone keywords such as "UTC" (Coordinated Universal Time) or "EST" (Eastern Standard Time). For TIMEX-es like *6:00 pm Saturday*, after normalizing *6:00 pm* to *T18:00*, we will see that we have missed *Saturday*. In this case, we normalize *Saturday* using the date rules above to get an anchor time, say *2017-05-06*. Combining them we have the final normalization value as *2017-05-06T18:00*.

Another type of TIMEX is called SET, which represents recurrent events (e.g., *every Monday*). If the TIMEX is a special adverb like *weekly* or *daily*, we directly normalize it and return the normalized values. Let "frequency" be words or phrases such as *once*, *twice*, *3 times*, and *3 days*; "quant" be words or phrases such as *every*,

each, and *per*; and "unit" be *day*, *week*, and *month*, etc. Then SET TIMEX-es are detected by extracting words or phrases with the format of "quant unit" (e.g., *every day*), "frequency number unit" (e.g., *once a day*), "quant number unit" (e.g., *every one day*), or "frequency quant number unit" (e.g., *once every two days*).

Finally DURATION are TIMEX-es that represent a period of time (e.g., *five days*). There are a few special cases for DURATION based on the annotation guidelines. First, since fractions such as *half a week* are allowed, we will use a more fine-grained unit "day" and then convert it to 3 days. This check of fractions is performed recursively until no fractions are detected. Then we normalize any DURATION with an explicit number that is not fractional (e.g., *3 weeks*). For phrases without an explicit number, we normalize them based on whether they are in plural form or not: If the time unit is in its singular form, the number is implicitly "1"; otherwise, we will use "X" as a placeholder for the number which indicates ambiguity (e.g., *weeks*).

3.3 Experiments

We use three newswire datasets provided in the TempEval3 workshop [3]: TimeBank, AQUAINT, and Platinum. The statistics of this corpus is given in Table 3.1. We used the original train/test split in our experiments: TimeBank and AQUAINT were for training (256 articles), and Platinum was for testing (20 articles). TemporalChunker took 10% of the train set as the development set.

Table 3.1: Statistics of the datasets provided in the TempEval3 workshop [3].

Corpus	# of Tokens	# of TIMEX	# of Documents
TimeBank	61k	1414	183
AQUAINT	34k	605	73
Platinum	6k	138	20

We adopt the following evaluation metrics for extraction and normalization. For

TIMEX extraction, we use precision P and recall R as follows.

$$P = \frac{|\text{Sys}_{\text{timex}} \cap \text{Ref}_{\text{timex}}|}{|\text{Sys}_{\text{timex}}|},$$
$$R = \frac{|\text{Sys}_{\text{timex}} \cap \text{Ref}_{\text{timex}}|}{|\text{Ref}_{\text{timex}}|},$$

where $\text{Sys}_{\text{timex}}$ means the set of all TIMEX-es predicted by the proposed TIMEX extraction system, $\text{Ref}_{\text{timex}}$ the set of all gold TIMEX-es in the dataset, and $|\cdot|$ is the cardinality of the set. As a standard, we also show the harmonic mean of precision and recall,

$$F_1 = \frac{2}{1/P + 1/R} = \frac{2PR}{P+R}.$$

As for TIMEX normalization, we use two metrics. When directly applied to gold TIMEX-es in the dataset, the proposed normalization system is evaluated on the accuracy metric, which is the number of those correct normalizations over the total number of TIMEX-es. When evaluating the end-to-end system performance where system TIMEX extraction is used, we need to take into account both the extraction performance and the normalization performance. That leads to the following modified precision and recall:

$$P = \frac{|\{\forall x | x \in (\text{Sys}_{\text{timex}} \cap \text{Ref}_{\text{timex}}) \wedge \text{Sys}_{\text{norm}}(x) == \text{Ref}_{\text{norm}}(x)\}|}{|\text{Sys}_{\text{timex}}|},$$
$$R = \frac{|\{\forall x | x \in (\text{Sys}_{\text{timex}} \cap \text{Ref}_{\text{timex}}) \wedge \text{Sys}_{\text{norm}}(x) == \text{Ref}_{\text{norm}}(x)\}|}{|\text{Ref}_{\text{timex}}|},$$
$$F_1 = \frac{2}{1/P + 1/R} = \frac{2PR}{P+R},$$

where $\text{Sys}_{\text{norm}}(x)$ and $\text{Ref}_{\text{norm}}(x)$ are the normalized value of x by the proposed system and by the data annotation, respectively. Note that if x is not a correct TIMEX in $\text{Ref}_{\text{timex}}$, then $\text{Sys}_{\text{norm}}(x) == \text{Ref}_{\text{norm}}(x)$ will definitely not be true, thus penalizing both precision and recall.

Table 3.2 evaluates the proposed TIMEX component, comparing with state-of-the-art systems. The "normalization" and "end-to-end" columns were evaluated based on gold TIMEX extraction and system TIMEX extraction, respectively. The fact that the proposed method had the best extraction F_1 and normalization accuracy but not the best end-to-end performance is due to our mixed strategy: TIMEX-es extracted by our learning-based TemporalChunker sometimes cannot be normalized correctly by our rule-based normalizer. This phenomenon is more severe in the proposed method comparing to systems that are consistently rule-based or learning-based in both extraction and normalization. However, the computational efficiency is improved significantly by reducing the runtime of the second fastest, HeidelTime, by more than 50%.

Table 3.2: Performance of the proposed TIMEX component compared with several state-of-the-art systems on the Platinum dataset from the TempEval3 workshop [3]. The "extraction" and "normalization" columns are the two intermediate steps. "Normalization" was evaluated given gold extraction, while "end-to-end" means system extraction was used. Runtimes were evaluated under the same setup.

TIMEX Systems	Extraction			Normalization	End-to-end	Runtime
	P	R	F_1	Accuracy	F_1	Seconds
HeidelTime [4]	84.0	79.7	81.8	78.1†	78.1	18
SUTime [5]	80.0	81.1	80.6	69.8†	69.8	16
UWTime [8]	**86.7**	80.4	83.5	84.4	**82.7**	400
Proposed	86.5	**83.3**	**84.9**	**84.7**	76.8	**7**

†HeidelTime and SUTime have no clear-cut between extraction and normalization, so even if gold TIMEX chunks are fed in, their extraction step cannot be skipped.

CHAPTER 4

TEMPORAL RELATION UNDERSTANDING

The second topic of this thesis is temporal relation (TempRel) understanding, which is complementary to the work on Timex understanding in Chapter 3. The TempRel task is to determine which event happens temporally earlier or later than the other (or other more fine-grained temporal relations such as *overlapping*). It has long been a challenging problem which previous systems did not do well, and that is why this thesis work emphasizes TempRel. This chapter will first explain some useful concepts such as temporal graph, the label space of TempRel, and the evaluation metrics in Sec. 4.1. At a high level, this thesis work has contributed to TempRel understanding via exploiting three inherent structures of time, namely: the transitivity structure of time which leads to a structured learning approach (Sec. 4.2), the probabilistic structure of time in the format of common sense which leads to a useful knowledge base (Sec. 4.3), and the sentential structure of time which leads to a new data annotation scheme (Sec. 4.4). At the end of this chapter, we will also put all three types of structure into a unified framework and show the combined improvement (Sec. 4.5).

4.1 Preliminaries

4.1.1 Temporal Graphs

In Sec. 2.1, we have introduced the definition of events. In practice, when we are given a document or multiple documents, there are multiple events. When all the

events are considered, we get a graph: where the nodes represent events and the edges TempRels. We hereafter call such graphs *temporal graphs*. Figure 4.1 shows the temporal graph representation of Example 9. The TempRel task can thus be modeled as a graph extraction problem.

A valid temporal graph needs to satisfy the following two properties:

1. *Symmetry*. For example, if A is *before* B, then B must be *after* A.
2. *Transitivity*. For example, if A is *before* B and B is *before* C, then A must be *before* C.

Due to the symmetry property, we can use a single and directed edge to represent the TempRel between two nodes; as in Example 9, *e11:hurt* is *after e10:ripping*, but in Fig. 4.1, we are safe to use a single "*before*" relation edge pointing from *e10:ripping* to *e11:hurt*.

Example 9:
. . . tons of earth (*e9:cascaded*) down a hillside, (*e10:ripping*) two houses from their foundations. No one was (*e11:hurt*), but firefighters (*e12:ordered*) the evacuation of nearby homes and said they'll (*e13:monitor*) the shifting ground. . . .

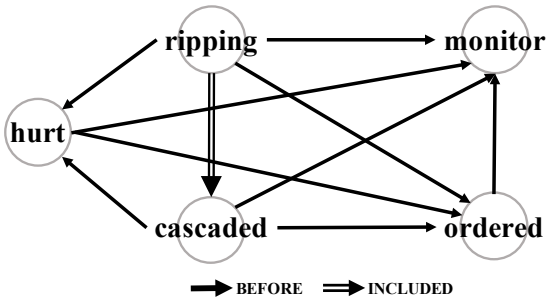


Figure 4.1: The temporal graph of Example 9, where the nodes represent events, and the edges represent the TempRels among those events.

The transitivity property, however, needs to be treated with caution. Specifically, it interrelates all the nodes in a graph, so the decision of one individual TempRel

depends on, or is even dictated by, TEMPRELs among other nodes. The transitivity property has two further implications. First, the TEMPREL *annotation process* is very challenging even for humans due to this global consideration since it potentially requires an annotation decision to be based on the entire article. The data annotation task hence needs to be designed very carefully. For instance, Fig. 4.2 is the actual annotations provided in the TempEval3 workshop and many TEMPRELs were mistakenly left unannotated, which is partly due to the excessive burden on annotators to make globally coherent annotations. The second implication is that TEMPREL systems also need to produce temporal graphs that respect this transitivity property. There has been much prior work incorporating global considerations in the *inference* phase, and we have also investigated ways to incorporate global considerations in the *learning* phase in Sec. 4.2.

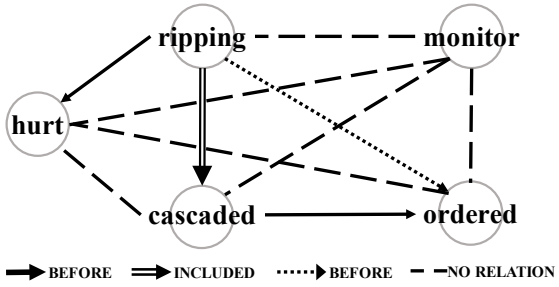


Figure 4.2: The human-annotation for Example 9 provided in the TempEval3 dataset, where many TEMPRELs are missing due to the annotation difficulty. Solid lines: original human annotations. Dotted lines: TEMPRELs inferred from solid lines. Dashed lines: missing relations.

In addition to the two properties, another important issue of temporal graph extraction is the definition of its nodes (or events). Generally speaking, an event is considered to be an action associated with corresponding participants involved in this action. However, as we pointed out in Sec. 2.1, the definition of events is often limited to a set of predefined types. In this thesis work so far, we have been following the event definition of TempEval3 [3] (which itself followed TimeBank [89]), where the limitation to predefined types is lifted; instead, all terms for situations that hap-

pen or occur are considered events, except for generic[1] and temporally static events.[2] If not stated otherwise, this thesis assumes gold events and only the TempRels need to be inferred.

4.1.2 Temporal Relation Labels

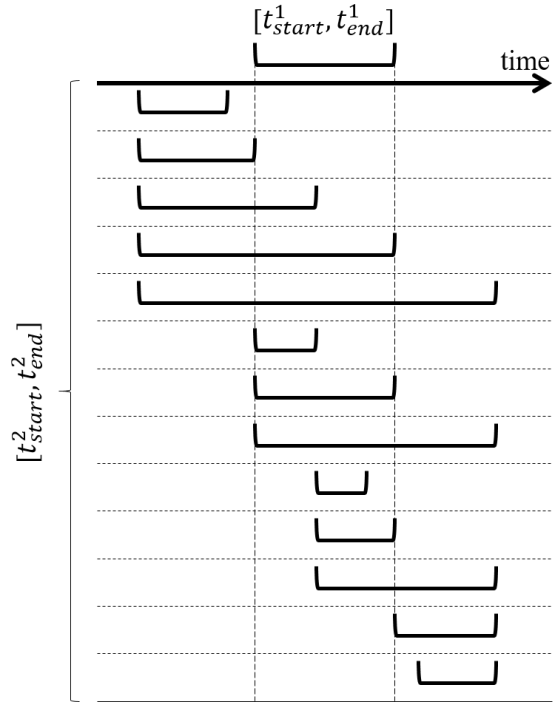


Figure 4.3: Thirteen possible relations between two events whose timescopes are $[t^1_{start}, t^1_{end}]$ and $[t^2_{start}, t^2_{end}]$ (from top to bottom): *after, immediately after, after and overlap, ends, included, started by, equal, starts, includes, ended by, before and overlap, immediately before* and *before*.

The TempRel between two time points is often straightforward: *before*, *after*, and *simultaneous*. Following Allen (1984) [90], existing works often represent the time scope of an event by an interval, $[t_{start}, t_{end}]$ (i.e., the start- and end-point of an event).

[1] Jews are prohibited from *killing* one another.
[2] New York *is* on the east coast.

Generally speaking, comparing two time intervals is the same as comparing four time points, which results in 13 different TempRel labels (see Fig. 4.3). However, existing works usually use a reduced set of these 13 labels. For instance, in Bethard et al. (2007) [91] and Do et al. (2012) [92], only 4 specific relations were considered: *before*, *after*, *overlap* and *simultaneous*; TimeBank-Dense [93] uses 5 relations: *before*, *after*, *includes*, *is included* and *simultaneous*. We think that people use a reduced set instead of the original 13 mainly due to the following two reasons.

1. The non-uniform distribution of all the 13 labels makes it difficult to separate low-frequency ones from the others (see Table 1 in [94]). For example, labels such as *immediately before* or *immediately after*, albeit possible, rarely exist in practice. As a result, intentionally omitting the rarely existing labels in the label set of a system often leads to better performances.

2. Due to ambiguity in natural language, the subtask of differentiating between *before* and *immediately before* may be not well-defined [95]. In Example 10, it is disputable whether *e14:locked* is *before* or *immediately before* due to the ambiguity between $t_{end}^{''locked''}$ and $t_{start}^{''left''}$. In addition, the granularity that the user cares about also affects the decision here. As a result, intentionally reducing these confusing labels often leads to better annotation agreement levels.

Example 10: I (*e14:locked*) the door and (*e15:left*) the place.

In my thesis work, we follow the reduced set used by TimeBank-Dense (i.e., *before*, *after*, *includes*, *is included* and *equal*). However, in Sec. 4.4, where we introduce our newly collected TempRel dataset, we switch to only focus on the TempRel between the start-points of events, and on that particular dataset, the labels are changed to *before*, *after*, and *equal*. Additionally, in both the aforementioned [91, 92, 93, 95] and this thesis work, an extra label called *vague* or *none* is also included as another relation type when the TempRel is not clear and no single relations can be convincingly assigned to it. In Example 11, it is unclear whether *e16:ate* or *e17:drank* happened first. Vagueness has long been an issue for the TempRel

task: For human annotators, it leads to confusion and lowers the inter-annotator agreement (IAA) levels; for systems, it is very difficult to tell if a TEMPREL is *vague* or not.

> **Example 11:** I (*e16:ate*) a burger and (*e17:drank*) a bottle of juice for lunch today.

4.1.3 Evaluation Metrics

There are three commonly-used evaluation metrics for the TEMPREL extraction task. The first metric is classification *accuracy* (Acc), which is the ratio of the number of correct TEMPRELs to the total number of predictions. The second metric is to view this task as a general relation extraction task, treat the label of *vague* for TEMPREL as *no relation*, and then compute the precision, recall, and F_1 accordingly.

The third metric, the temporal awareness score, came into use since the TempEval3 workshop [3], which involves graph closure and reduction on top of the second metric. Specifically, let G_{sys} and G_{ref} be two temporal graphs from the system prediction and the reference (e.g., the ground truth), respectively. The precision and recall in the temporal awareness setup are defined as follows.

$$P = \frac{|G^-_{\text{sys}} \cap G^+_{\text{ref}}|}{|G^-_{\text{sys}}|}, \; R = \frac{|G^-_{\text{ref}} \cap G^+_{\text{sys}}|}{|G^-_{\text{ref}}|},$$

where G^+ is the closure of graph G, G^- is the reduction of G, "$\cap$" is the intersection between TEMPRELs in two graphs, and $|G|$ is the number of TEMPRELs in G (note that *vague* relations are not counted). The temporal awareness metric better captures how "useful" a temporal graph is, for example, if system 1 produces *ripping* is *before hurt* and *hurt* is *before monitor*, and system 2 adds *ripping* is *before monitor* on top of system 1. Since system 2 is simply a transitive closure of system 1, they would have the same evaluation scores in the temporal awareness setup.

Take the confusion matrix in Fig. 4.4 for example. The three metrics used in this thesis are

1. Accuracy. $Acc = (C_{b,b} + C_{a,a} + C_{e,e} + C_{v,v})/S$.

2. Precision, recall, and F_1. $P = (C_{b,b}+C_{a,a}+C_{e,e})/S_1$, $R = (C_{b,b}+C_{a,a}+C_{e,e})/S_2$, and $F_1 = 2PR/(P + R)$.

3. Awareness score F_{aware}. Before calculating precision, perform a graph closure on the gold temporal graph and a graph reduction on the predicted temporal graph. Similarly, before calculating recall, perform a graph reduction on the gold temporal graph and a graph closure on the predicted temporal graph. Finally, compute the F_1 score based on this revised precision and recall. Since graph reduction and closure are involved in computing this metric, the temporal graphs all need to satisfy the global transitivity constraints of temporal relations (e.g., if A happened *before* B, and B happened *before* C, then C cannot be *before* A if we want to use this metric).

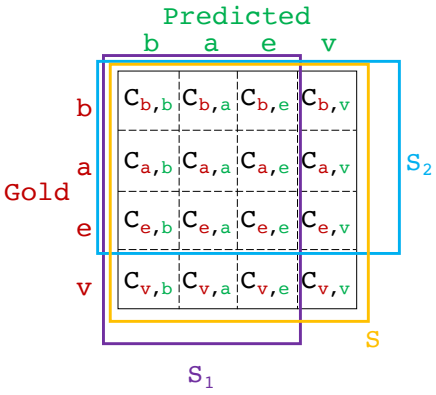


Figure 4.4: An example confusion matrix, where the four labels are *before* (b), *after* (a), *simultaneous* (e), and *vague* (v), respectively. Note this is only for illustration purpose and the label set can be different in practice. The variables, S, S_1, and S_2, are the summation of all the numbers in the corresponding area. This figure is better viewed in color.

4.2 Structured Learning for TempRel Extraction

4.2.1 Related Work

In order to solve the graph extraction problem in the TempRel task, early attempts [94, 96, 91, 97] studied *local* methods. That is, look at each pair of nodes and make decisions irrespective of edges between other pairs, during which both the learning and inference are *local*. Some of the state-of-the-art methods, including ClearTK [98], UTTime [99], and NavyTime [100], use better designed rules or more advanced features such as syntactic tree paths, but are still *local* in this context. A disadvantage is that decisions made locally may be globally inconsistent (i.e., the symmetry and/or transitivity constraints are not satisfied for the entire temporal graph).

Integer linear programming (ILP) methods, which were introduced to solve inference problems in NLP [74], have been used for the TempRel task in order to enforce global consistency in several works including [101, 50, 92]. They formulated temporal graph extraction as an ILP and showed that it improved over local methods for densely connected graphs. As we mentioned in Sec. 2.2.4, since these methods perform inference ("I") on top of pre-trained local classifiers ("L"), they are often referred to as L+I [80]. In another state-of-the-art method, CAscading EVent Ordering (CAEVO) [95], some hand-crafted rules and machine learned classifiers (called sieves therein) form a pipeline. Global consistency is enforced by inferring all possible relations before passing the graph to the next sieve. This best-first architecture is conceptually similar to L+I but the inference is greedy, similar to [102, 97]. In other words, these methods have all successfully incorporated the transitivity structure in the inference phase.

While it is clear that the transitivity structure of temporal graphs requires global considerations when solving the TempRel task, all the aforementioned methods depend on classifiers that are *learned* locally without taking structural information into account. Although L+I methods impose global constraints in the *inference* phase, we argue that global considerations are necessary in the *learning* phase as well, which falls into the category of structured learning.

In parallel to the work presented here, [103] also proposed a structured learning approach to extracting the TempRels. Their work mainly focused on the medical domain based on the Clinical TempEval workshop [104, 105, 106], so their work provides additional evidence that structured learning is a suitable choice for the TempRel task. More importantly, they compared structured learning to local baselines, while we find that the comparison between structured learning and L+I is more interesting and important for understanding the effect of global considerations in the learning phase. Given the transitivity property that valid temporal graphs possess, the TempRel extraction problem is a structured prediction problem. In this section, we explain our approach to both learning and inference.

Another line of work that takes advantage of the structure of time is called *temporal dependency structure* (Zhang and Xue [107, 108]). The same authors have also proposed corresponding neural parsers for this structure [109]. This temporal dependency structure treats a temporal graph as a dependency parsing tree where each dependency relation corresponds to an instance of temporal anaphora where the antecedent is the parent and the anaphor is the child. The structure is different to the transitivity structure exploited here.

4.2.2 Inference via Integer Linear Programming

Since inference is an important step in structured learning schemes, we first introduce the inference algorithm via ILP. In a temporal graph with n edges, let $\phi_i \in \mathcal{X} \subseteq \mathbb{R}^d$ be the extracted d-dimensional feature and $y_i \in \mathcal{Y}$ be the TempRel for the i-th edge, $i = 1, 2, \ldots, n$, where $\mathcal{Y} = \{r_j\}_{j=1}^{6}$ is the label set for the six TempRels we use, i.e., *before*, *after*, *includes*, *is_included*, *simultaneous*, and *vague*. Moreover, let $\mathbf{x} = \{\phi_1, \ldots, \phi_n\} \in \mathcal{X}^n$ and $\mathbf{y} = \{y_1, \ldots, y_n\} \in \mathcal{Y}^n$ be more compact representations of all the features and labels in this temporal graph. Given the weight vector $\mathbf{w}_r$ of a linear classifier trained for relation $r \in \mathcal{Y}$ (i.e., using the one-vs-all scheme), the global inference step is to solve the following constrained optimization problem:

$$\hat{\mathbf{y}} = \arg \max_{\mathbf{y} \in \mathcal{C}(\mathcal{Y}^n)} f(\mathbf{x}, \mathbf{y}), \tag{4.1}$$

where $\mathcal{C}(\mathcal{Y}^n) \subseteq \mathcal{Y}^n$ constrains the temporal graph to be symmetrically and transitively consistent, and $f(\mathbf{x}, \mathbf{y})$ is the soft-max scoring function:

$$f(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} f_{y_i}(\phi_i) = \sum_{i=1}^{n} \frac{e^{\mathbf{w}_{y_i}^T \phi_i}}{\sum_{r \in \mathcal{Y}} e^{\mathbf{w}_r^T \phi_i}}.$$

Specifically, $f_{y_i}(\phi_i)$ is the probability of the i-th edge having relation y_i. $f(\mathbf{x}, \mathbf{y})$ is simply the sum of these probabilities over all the edges in a temporal graph, which we think of as the confidence of assigning $\mathbf{y} = \{y_1, ..., y_n\}$ to this temporal graph, and it therefore needs to be maximized in Eq. (4.1).

Note that when $\mathcal{C}(\mathcal{Y}^n) = \mathcal{Y}^n$, Eq. (4.1) can be solved for each $\hat{y}_i$ independently, which is what the so-called local methods do. When $\mathcal{C}(\mathcal{Y}^n) \neq \mathcal{Y}^n$, Eq. (4.1) cannot be decoupled for each $\hat{y}_i$ and is usually formulated as an ILP problem [74, 50, 92]. Specifically, let $\mathcal{I}_r(ij) \in \{0, 1\}$ be the indicator function of relation r for node i and node j and $f_r(ij) \in [0, 1]$ be the corresponding soft-max score. Then the ILP objective for global inference is formulated as follows.

$$\begin{aligned} \hat{\mathcal{I}} = \underset{\mathcal{I}}{\operatorname{argmax}} & \textstyle\sum_{i<j} \sum_{r \in \mathcal{Y}} f_r(ij) \mathcal{I}_r(ij) \\ \text{s.t.} \quad & \underbrace{\textstyle\Sigma_r \mathcal{I}_r(ij) = 1,}_{\text{(uniqueness)}} \\ & \underbrace{\mathcal{I}_{r_1}(ij) + \mathcal{I}_{r_2}(jk) - \textstyle\Sigma_{m=1}^{N} \mathcal{I}_{r_3^m}(ik) \leq 1,}_{\text{(transitivity)}} \end{aligned} \tag{4.2}$$

for all distinct nodes i, j, and k, where N is the number of possible relations for r_3 when r_1 and r_2 are true. The formulation in Eq. (4.2) is different from previous work [50, 92]. Previously, transitivity constraints were formulated as $\mathcal{I}_{r_1}(ij) + \mathcal{I}_{r_2}(jk) - \mathcal{I}_{r_3}(ik) \leq 1$, which is a special case when $N = 1$ and can be understood as "r_1 and r_2 determine a single r_3". Imagine if both r_1 and r_2 are true, then the only way to satisfy this constraint is to have r_3 be true as well. However, it was overlooked that, although some r_1 and r_2 cannot uniquely determine r_3, they can still constrain the set of labels that r_3 can take. For example, as shown in Fig. 4.5, when r_1=*before* and r_2=*is_included*, r_3 is not determined but we know that $r_3 \in \{$*before*, *is_included*$\}$.

This information can be easily exploited by allowing $N > 1$. Note that despite this difference, this optimization problem (4.2) can still be solved using off-the-shelf ILP packages such as GUROBI [75].

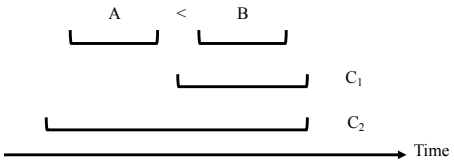


Figure 4.5: When *A is before B* and *B is_included in C*, *A* can either be *before* C_1 or *is_included in* C_2. We propose to incorporate this via the transitivity constraints for Eq. (4.2).

Let $r_3 \in \text{Trans}(r_1, r_2)$ be the set comprised of all the TempRel labels that do not conflict with r_1 and r_2. Using this notation, the transitivity constraint in Eq. (4.2) can be rewritten as

$$\mathcal{I}_{r_1}(ij) + \mathcal{I}_{r_2}(jk) - \sum_{r_3 \in \text{Trans}(r_1, r_2)} \mathcal{I}_{r_3}(ik) \leq 1.$$

Imagine if both r_1 and r_2 are true, then the only way to satisfy this constraint is to select r_3 from the set of $\text{Trans}(r_1, r_2)$.

The construction of $\text{Trans}(r_1, r_2)$ necessitates a clearer definition of the label set of TempRels, the importance of which is often overlooked by existing methods. We have used $\mathcal{Y}$ as the label set used here, i.e., *before*, *after*, *includes*, *is_included*, *simultaneous*, and *vague*. For notation convenience, we denote them as $\mathcal{Y} = \{b, a, i, ii, s, v\}$. As we have introduced in Sec. 4.1.2, existing approaches all followed the interval representation of events [90], which originally yields 13 TempRel labels (Fig. 4.3; let $\tilde{\mathcal{Y}}$ be the 13 labels plus *vague*). Many systems used a reduced set, for example, our $\mathcal{Y}$ here. However, there has been limited discussion in the literature on how to interpret the reduced relation types. For example, is the "*before*" in $\mathcal{Y}$ exactly the same as the "*before*" in the original set ($\tilde{\mathcal{Y}}$) (as shown on the left-hand-side of Fig. 4.6), or is it a combination of multiple relations in $\tilde{\mathcal{Y}}$ (the right-hand side of Fig. 4.6)? We have tried both reduction schemes in Fig. 4.6, where scheme 1 ignores low frequency

labels directly and scheme 2 absorbs low frequency ones into their temporally closest labels. The two schemes barely have differences when a system only looks at a single pair of mentions at a time (this might explain the lack of discussion over this issue in the literature), but they lead to different Trans(r_1, r_2) sets and this difference can be magnified when the problem is solved jointly and when the label distribution changes across domains. To completely cover the 13 relations, we adopt scheme 2 in Fig. 4.6 in this work.

The resulting transitivity relations are shown in Table 4.1. The top part of Table 4.1 is a compact representation of three generic rules; for instance, Line 1 means that the labels themselves are transitive. Note that during human annotation, if an annotator looks at a pair of events and decides that multiple well-defined relations can exist, he/she labels it *vague*; also, when aggregating the labels from multiple annotators, a label will be changed to *vague* if the annotators disagree with each other. In either case, *vague* is chosen to be the label when a single well-defined relation cannot be uniquely determined by the contextual information. This explains why a *vague* relation (v) is always added in Table 4.1 if more than one label in Trans(r_1, r_2) is possible. As for Lines 6, 9-11 in Table 4.1 (where *vague* appears in Column r_2), Column Trans(r_1,r_2) was designed in such a way that r_2 cannot be uniquely determined through r_1 and Trans(r_1,r_2). For instance, r_1 is *after* on Line 9, if we further put *before* into Trans(r_1,r_2), then r_2 would be uniquely determined to be *before*, conflicting with r_2 being *vague*, so *before* should not be in Trans(r_1,r_2).

4.2.3 Learning via Structured Perceptron

With the inference solver defined above, we propose to use the structured perceptron [110] as a representative structured learning algorithm for TempRel extraction. Specifically, let $\mathcal{L} = \{\mathbf{x}_k, \mathbf{y}_k\}_{k=1}^{K}$ be the labeled training set of K instances (usually documents). The structured perceptron training algorithm for this problem is shown in Algorithm 1. The Illinois-SL package [111] was used in our experiments for its structured perceptron component. In terms of the features used in this work, we adopt the same set of features designed in Sec. 3.1 of [92].

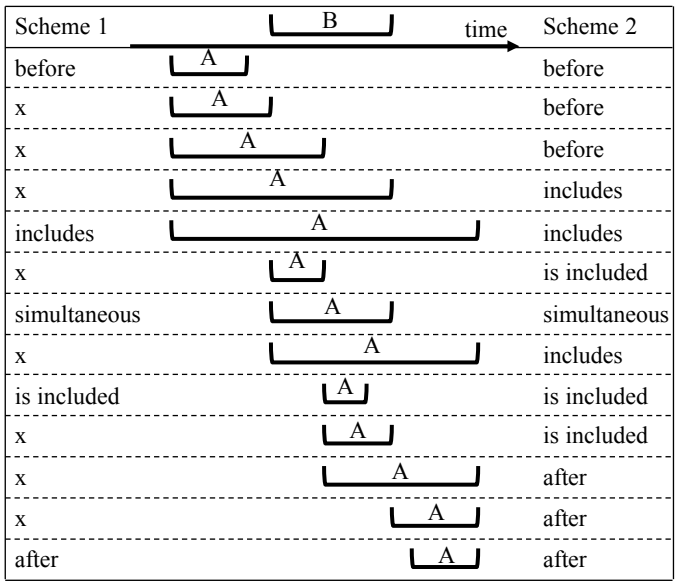


Figure 4.6: **Two possible interpretations to the label set of** $\mathcal{Y} = \{b, a, i, ii, s, v\}$ for the temporal relations between (A, B). "x" means that the label is ignored. Brackets represent time intervals along the time axis.

In Algorithm 1, Line 5 is the inference step as in Eq. (4.1) or (4.2). If there was only one pair of events in each instance (thus no structure to take advantage of), Algorithm 1 would reduce to the conventional perceptron algorithm and Line 5 simply chooses the top scoring label. With a structured instance instead, Line 5 becomes slower to solve, but it can provide valuable information so that the perceptron learner is able to look further at other labels rather than an isolated pair. For example in Example 9 and Fig. 4.1, the fact that (ripping,ordered)=*before* is established through two other relations: 1) *ripping* is an adverbial participle and thus *included* in *cascaded* and 2) *cascaded* is *before ordered*. If (ripping,ordered)=*before* is presented to a local learning algorithm without knowing its predictions on (ripping,cascaded) and (cascaded,ordered), then the model either cannot support it or overfits it. In structured perceptron, however, if the classifier was correct in deciding (ripping,cascaded) and (cascaded,ordered), then (ripping,ordered) would be correct automatically due to structural constraints, and would not contribute to updating the classifier.

Table 4.1: **Transitivity relations** based on the label set reduction scheme 2 in Fig. 4.6. If $(m_1, m_2) \mapsto r_1$ and $(m_2, m_3) \mapsto r_2$, then the relation of (m_1, m_3) must be chosen from Trans(r_1, r_2), $\forall m_1$, m_2, $m_3 \in \mathcal{M}$. The top part of the table uses r to represent generic rules compactly. Notations: before (**b**), after (**a**), includes (**i**), is_included (**ii**), simultaneously (**s**), vague (**v**), and $\bar{r}$ represents the reverse relation of r.

No.	r_1	r_2	Trans(r_1, r_2)
1	r	r	r
2	r	**s**	r
3	r_1	r_2	$\overline{\text{Trans}(\bar{r}_2, \bar{r}_1)}$
4	**b**	**i**	**b**, **i**, **v**
5	**b**	**ii**	**b**, **ii**, **v**
6	**b**	**v**	**b**, **i**, **ii**, **v**
7	**a**	**i**	**a**, **i**, **v**
8	**a**	**ii**	**a**, **ii**, **v**
9	**a**	**v**	**a**, **i**, **ii** ,**v**
10	**i**	**v**	**b**, **a**, **i**, **v**
11	**ii**	**v**	**b**, **a**, **ii**, **v**

Algorithm 1: Structured perceptron algorithm for TempRels

Input: Training set $\mathcal{L} = \{\mathbf{x}_k, \mathbf{y}_k\}_{k=1}^K$, learning rate λ

Perform graph closure on each $\mathbf{y}_k$

Initialize $\mathbf{w}_r = \mathbf{0}$, $\forall r \in \mathcal{Y}$

while *convergence criteria not satisfied* **do**

  Shuffle the examples in $\mathcal{L}$

  foreach $(\mathbf{x}, \mathbf{y}) \in \mathcal{L}$ **do**

    $\hat{\mathbf{y}} = \arg\max_{\mathbf{y} \in \mathcal{C}} f(\mathbf{x}, \mathbf{y})$

    if $\hat{\mathbf{y}} \neq \mathbf{y}$ **then**

      $\mathbf{w}_r = \mathbf{w}_r + \lambda(\sum_{i:\mathbf{y}_i = r} \phi_i - \sum_{i:\hat{\mathbf{y}}_i = r} \phi_i)$, $\forall r \in \mathcal{Y}$

return $\{\mathbf{w}_r\}_{r \in \mathcal{Y}}$

4.2.4 Learning via CoDL

The scarcity of training data and the difficulty in annotation have long been a bottleneck for temporal processing systems. Given the inherent global constraints in temporal graphs, we propose to perform semi-supervised structured learning using the constraint-driven learning (CoDL) algorithm [112, 113], as shown in Algorithm 2, where the function "Learn" in Lines 2 and 9 represents any standard learning algorithm (e.g., perceptron, SVM, or even structured perceptron; here we used the averaged perceptron [114]) and subscript "r" means selecting the learned weight vector for relation $r \in \mathcal{Y}$. CoDL improves the model learned from a small amount of labeled data by repeatedly generating feedback through labeling unlabeled examples, which is in fact a semi-supervised version of IBT. Experiments show that this scheme is indeed helpful for solving this problem.

Algorithm 2: Constraint-driven learning algorithm

Input: Labeled set $\mathcal{L}$, unlabeled set $\mathcal{U}$, weighting coefficient γ

1 Perform closure on each graph in $\mathcal{L}$
2 Initialize $\mathbf{w}_r = \text{Learn}(\mathcal{L})_r, \forall\ r \in \mathcal{Y}$
3 **while** *convergence criteria not satisfied* **do**
4 $\mathcal{T} = \emptyset$
5 **foreach** $\mathbf{x} \in \mathcal{U}$ **do**
6 $\hat{\mathbf{y}} = \arg\max_{\mathbf{y} \in \mathcal{C}} f(\mathbf{x}, \mathbf{y})$
7 Perform graph closure on $\hat{\mathbf{y}}$
8 $\mathcal{T} = \mathcal{T} \cup \{(\mathbf{x}, \hat{\mathbf{y}})\}$
9 $\mathbf{w}_r = \gamma \mathbf{w}_r + (1 - \gamma)\text{Learn}(\mathcal{T})_r, \forall\ r \in \mathcal{Y}$
10 **return** $\{\mathbf{w}_r\}_{r \in \mathcal{Y}}$

4.2.5 Missing Annotations

Since even human annotators find it difficult to annotate temporal graphs, many of the TempRels are left unspecified by annotators (compare Fig. 4.2 to Fig. 4.1). While some of these missing TempRels can be inferred from existing ones, the vast

majority still remain unknown as shown in Table 4.2. Despite the existence of denser annotation schemes (e.g., Cassidy et al. (2014) [93]), the TempRel annotation task is quadratic in the number of nodes, and it is practically infeasible to annotate complete graphs. Therefore, the problem of identifying these unknown relations in training and test is an important issue that dramatically hurts existing methods.

Table 4.2: Categories of Event-Event TempRels in the TE3 Platinum dataset. Among all pairs of events, 98.2% of them are left unspecified by the annotators. Graph closure can automatically add 8.7%, but most of the event pairs are still unknown.

Type		#TempRel	%
Annotated		582	1.8
Missing	Inferred	2840	8.7
	Unknown	29240	89.5
Total		32662	100

We could simply use these unknown pairs (or some filtered version of them) to design rules or train classifiers to identify whether a TempRel is *vague* or not. However, we propose to exclude both the unknown pairs and the *vague* classifier from the training process – by changing the structured loss function to ignore the inference feedback on *vague* TempRels (see Line 8 in Algorithm 1 and Line 9 in Algorithm 2). The reasons are discussed below.

First, it is believed that a lot of the unknown pairs are not really *vague* but rather pairs that the annotators failed to look at [91, 93, 95]. For example, (cascaded, monitor) should be annotated as *before* but is missing in Fig. 4.2. It is hard to exclude such noise in the data during training. Second, compared to the overwhelmingly large number of unknown TempRels (89.5% as shown in Table 4.2), the scarcity of non-vague TempRels makes it hard to learn a good *vague* classifier. Third, *vague* is fundamentally different from the other relation types. For example, if a *before* TempRel can be established given a sentence, then it always holds as *before* regardless of other events around it, but if a TempRel is *vague* given a sentence, it may still change to other types afterwards if a connection can later be established through other nodes from the context. This distinction emphasizes that *vague* is a

consequence of lack of background/contextual information, rather than a concrete relation type to be trained on. Fourth, without the *vague* classifier, the predicted temporal graph tends to become more densely connected, thus the global transitivity constraints can be more effective in correcting local mistakes [50].

However, excluding the local classifier for *vague* TempRels would undesirably assign non-vague TempRels to every pair of events. To handle this, we take a closer look at the *vague* TempRels. Note that a *vague* TempRel could arise in two situations if the annotators did not fail to look at it. One is that an annotator looks at this pair of events and decides that multiple relations can exist, and the other one is that two annotators disagree on the relation (similar arguments were also made in [93]). In both situations, the annotators first try to assign all possible relations to a TempRel, and then change the relation to *vague* if more than one can be assigned. This human annotation process for *vague* is different from many existing methods, which either identify the existence of a TempRel first (using rules or machine-learned classifiers) and then classify, or directly include *vague* as a classification label along with other non-vague relations.

We propose to mimic this mental process by a *post-filtering* method. Specifically, we take each TempRel produced by ILP and determine whether it is *vague* using its relative entropy (the Kullback-Leibler divergence) to the uniform distribution. Letting $\{r_m\}_{m=1}^{M}$ be the set of relations that the i-th pair of events can take, we filter the i-th TempRel given by ILP by:

$$\delta_i = \sum_{m=1}^{M} f_{r_m}(\phi_i) \log \left(M f_{r_m}(\phi_i)\right),$$

where $f_{r_m}(\phi_i)$ is the soft-max score of r_m, obtained by the local classifier for r_m. We then compare δ_i to a fixed threshold τ to determine the vagueness of this TempRel; we accept its originally predicted label if $\delta_i > \tau$, or change it to *vague* otherwise. Using relative entropy here is intuitively appealing and empirically useful as shown in the experiments section next.

4.2.6 Experiments

4.2.6.1 Datasets

The TempEval3 (TE3) workshop [3] provided the TimeBank (TB) [89], AQUAINT (AQ) [115], Silver (TE3-SV), and Platinum (TE3-PT) datasets, where TB and AQ are usually for training, and TE3-PT is usually for testing. The TE3-SV dataset is a much larger, machine-annotated and automatically-merged dataset based on multiple systems, with the intention to see if these "silver" standard data can help when included in training (although almost all participating systems saw performance drop with TE3-SV included in training).

Two popular augmentations on TB are the Verb-Clause TempRel dataset (VC) and Timebank-Dense dataset (TD). The VC dataset has specially annotated event pairs that follow the so-called Verb-Clause structure [91], which is usually beneficial to be included in training [3]. The TD dataset contains 36 documents from TB which were re-annotated using the dense event ordering framework proposed in [93]. The experiments included in this dissertation will involve the TE3 datasets as well as these augmentations. Therefore, some statistics on them are shown in Table 4.3.

Table 4.3: Facts about the datasets used in this section. Note that the column of TempRels only counts the non-vague TempRels. The TempRel annotations in TE3-SV is not used in this dissertation and its number is thus not shown.

Dataset	Doc	Tokens	Event	TempRel	Note
TB+AQ	256	100K	12K	12K	Training
VC	132	-	1.6K	0.9K	Training
TD	36	-	1.6K	5.7K	Training
TD-Train	22	-	1K	3.8K	Training
TD-Dev	5	-	0.2K	0.6K	Dev
TD-Test	9	-	0.4K	1.3K	Eval
TE3-PT	20	6K	0.7K	0.9K	Eval
TE3-SV	2.5K	666K	81K	-	Unlabeled

4.2.6.2 Baseline Methods

In addition to the state-of-the-art systems, another two baseline methods were also implemented for a better understanding of the proposed. The first is the regularized averaged perceptron (AP) [114] implemented in the LBJava package [116] and is a local method. On top of the first baseline, we performed global inference in Eq.(4.2), referred to as the L+I baseline (AP+ILP). Both of them used the same feature set (i.e., as designed in [92]) as in the proposed structured perceptron (SP) for fair comparisons. To clarify, SP is a training algorithm and its immediate outputs are the weight vectors $\{\mathbf{w}_r\}_{r\in\mathcal{Y}}$ for local classifiers. An ILP inference was performed on top of it to yield the final output, and we refer to it as "S+I" (i.e., structured learning+inference) methods.

Table 4.4: Temporal awareness scores on TE3-PT given gold event pairs. Systems that are significantly better than the previous rows are underlined (per McNemar's test with $p < 0.0005$). The last column shows the relative improvement in F1 score over AP-1, which identifies the source of improvement: 5.2% from additional training data, 9.3% (14.5%-5.2%) from constraints, and 10.4% from structured learning.

System	Method	P	R	F1	%
UTTime	Local	55.6	57.4	56.5	+5.0
AP-1	Local	56.3	51.5	53.8	0
AP-2	Local	58.0	55.3	56.6	+5.2
AP+ILP	L+I	62.2	61.1	61.6	+14.5
SP+ILP	S+I	**69.1**	**65.5**	**67.2**	**+24.9**

4.2.6.3 TE3 Task C - Relation Only

To show the benefit of using structured learning, we tested the scenario where the gold pairs of events that have a non-vague TempRel were known priori. This setup was a standard task presented in TE3 (Task C – Relation Only). UTTime [99] was the top system in this task in TE3. Since UTTime is not available to us, and its performance was reported in TE3 in terms of both Event-Event (EE) and Event-

Timex (ET) TempRels together, we also locally trained an ET classifier based on [92] and included its prediction only for fair comparisons.

UTTime is a local method and was trained on TB+AQ and tested on TE3-PT. We used the same datasets for our local baseline and its performance is shown in Table 4.4 under the name "AP-1". Note that the numbers reported below are the temporal awareness scores obtained from the official evaluation script provided in TE3. We can see that UTTime is about 3% better than AP-1 in the absolute value of F_1, which is expected since UTTime included more advanced features derived from syntactic parse trees. By adding the VC and TD datasets into the training set, we retrained our local baseline and achieved performance comparable to that of UTTime ("AP-2" in Table 4.4). On top of AP-2, a global inference step enforcing symmetry and transitivity constraints ("AP+ILP") can further improve the F_1 score by 9.3%, which is consistent with previous observations [50, 92]. SP+ILP further improved the performance in precision, recall, and F_1 significantly (per the McNemar's test [117, 118] with $p < 0.0005$), reaching an F_1 score of 67.2%. This meets our expectation that structured learning can be better when the local problem is difficult [80].

Table 4.5: Temporal awareness scores given gold events but with no gold pairs (TempEval3 Task C), which show that the proposed S+I methods outperformed state-of-the-art systems in various settings. The fourth column indicates the annotation sources used, with additional unlabeled dataset in the parentheses. The "Filters" column shows if the proposed post-filtering method (Sec. 4.2.5) was used. The last column is the relative improvement in F_1 score compared to baseline systems on line 1, 7, and 11, respectively. Systems that are significantly better than the "*"-ed systems are underlined (per McNemar's test with $p < 0.0005$).

No.	System	Method	Anno. (Unlabeled)	Testset	Filters	P	R	F1	%
1	ClearTK	Local	TB, AQ, VC, TD	TE3-PT	-	**37.2**	33.1	35.1	0
2	AP*	Local	TB, AQ, VC, TD	TE3-PT	-	35.3	37.1	36.1	+2.8
3	AP+ILP	L+I	TB, AQ, VC, TD	TE3-PT	-	35.7	35.0	35.3	+0.6
4	SP+ILP	S+I	TB, AQ, VC, TD	TE3-PT	-	32.4	45.2	37.7	+7.4
5	SP+ILP	S+I	TB, AQ, VC, TD	TE3-PT	post	33.1	**49.2**	39.6	+12.8
6	CoDL+ILP	S+I	TB, AQ, VC, TD (TE3-SV)	TE3-PT	post	35.5	46.5	**40.3**	**+14.8**
7	ClearTK*	Local	TB, VC	TE3-PT	-	**35.9**	38.2	37.0	0
8	SP+ILP	S+I	TB, VC	TE3-PT	post	30.7	**47.1**	37.2	+0.5
9	CoDL+ILP	S+I	TB, VC (TE3-SV)	TE3-PT	post	33.9	45.9	**39.0**	**+5.4**
10	ClearTK	Local	TD-Train	TD-Test	-	46.04	20.90	28.74	-
11	CAEVO*	L+I	TD-Train	TD-Test	-	**54.17**	39.49	45.68	0
12	SP+ILP	S+I	TD-Train	TD-Test	post	45.34	48.68	46.95	+3.0
13	CoDL+ILP	S+I	TD-Train (TE3-SV)	TD-Test	post	45.57	**51.89**	**48.53**	**+6.3**

4.2.6.4 TE3 Task C

In the first scenario, we knew in advance which TempRels existed or not, so the "post-filtering" method was not used when generating the results in Table 4.4. Here we test a more practical scenario, where we only know the events, but do not know which ones are related. This setup was Task C in TE3 and the top system was ClearTK [98]. Again, for fair comparison, we simply added the ET TempRels predicted by ClearTK. Moreover, 10% of the training data was held out for development. Corresponding results on the TE3-PT testset are shown in Table 4.5.

From lines 2-4, all systems see significant drops in performance if compared with the same entries in Table 4.4. It confirms our assertion that how to handle *vague* TempRels is a major issue for this TempRel extraction problem. The improvement of SP+ILP (line 4) over AP (line 2) was small and AP+ILP (line 3) was even worse than AP, which necessitates the use of a better approach towards *vague* TempRels. By applying the post-filtering method proposed in Sec. 4.2.5, we were able to achieve better performances using SP+ILP (line 5), which shows the effectiveness of this strategy. Finally, by setting $\mathcal{U}$ in Algorithm 2 to be the TE3-SV dataset, CoDL+ILP (line 6) achieved the best F_1 score with a relative improvement over ClearTK being 14.8%. Note that when using TE3-SV in this work, we did not use its annotations on TempRels.

In [3], we notice that the best performance of ClearTK was achieved when trained on TB+VC (line 7 is higher than its reported values in TE3 because of later changes in ClearTK), so we retrained the proposed systems on the same training set and results are shown in lines 8-9. In this case, the improvement of S+I over Local was small, which may be due to the lack of training data. Note that line 8 was still significantly different from line 7 per the McNemar's test, although there was only 0.2% absolute difference in F_1, which can be explained from their large differences in precision and recall.

4.2.6.5 Comparison with CAEVO

The proposed structured learning approach was further compared to a recent system, a CAscading EVent Ordering architecture (CAEVO) proposed in [95] (lines 10-13). We used the same training set and test set as CAEVO in the S+I systems. Again, we added the ET TempRels predicted by CAEVO to both S+I systems. In [95], CAEVO was reported on the straightforward evaluation metric including the *vague* TempRels, but the temporal awareness scores were used here, which explains the difference between line 11 in Table 4.5 and what was reported in [95].

ClearTK was reported to be outperformed by CAEVO on TD-Test [95], but we observe that ClearTK on line 10 was much worse even than itself on line 7 (trained on TB+VC) and on line 1 (trained on TB+AQ+VC+TD) due to the annotation scheme difference between TD and TB/AQ/VC. ClearTK was designed mainly for TE3, aiming for high precision, which is reflected by its high precision on line 10, but it does not have enough flexibility to cope with two very different annotation schemes. Therefore, we have chosen CAEVO as the baseline system to evaluate the significance of the proposed ones. On the TD-Test dataset, all systems other than ClearTK had better F_1 scores compared to their performances on TE3-PT. This notable difference (i.e., 48.53 vs 40.3) indicates the better quality of the dense annotation scheme that was used to create TD [93]. SP+ILP outperformed CAEVO and if additional unlabeled dataset TE3-SV was used, CoDL+ILP achieved the best score with a relative improvement in F_1 score being 6.3%.

So far, we have shown a structured learning approach to identifying TempRels in natural language text and show that it captures the global nature of this problem better than state-of-the-art systems do. In addition, the global nature of this problem gives rise to a better way of making use of the readily available unlabeled data, which further improves the proposed method. The improved performance on both TE3-PT and TD-Test, two differently annotated datasets, clearly shows the advantage of exploiting "structures" in time.

4.3 Injection of Human Prior Knowledge

As in many NLP tasks, one of the challenges in TempRel extraction is that it requires high-level prior knowledge; in this case, we care about the temporal order that events *usually* follow. In Example 12, we have deleted events from the original sentence. Rich temporal information is encoded in the events' names, and this often plays an indispensable role in making our decisions. As a result, it is very difficult even for humans to figure out the TempRels between those events. In the first paragraph of Example 12, it is difficult to understand what really happened without the actual event verbs; let alone the TempRels between them. In the second paragraph, things are even more interesting: if we had *e20:dislike* and *e21:stop*, then we would know easily that "I dislike" occurs *after* "they stop the column". However, if we had *e20:ask* and *e21:help*, then the relation between *e20* and *e21* would be reversed and *e20* is *before e21*. We are in urgent need of the event names to determine the TempRels. In Example 13, where we show the complete sentences, the task has become much easier for humans due to our prior knowledge. Motivated by these examples (which are in fact very common), we believe in the importance of such a prior knowledge in determining TempRels.

Example 12: Difficulty in understanding TempRels when event content is missing. Note that *e18* and *e19* have the same tense, and *e20* and *e21* have the same tense.
More than 10 people have (*e18:*), police said. A car (*e19:*) on Friday in the middle of a group of men playing volleyball.
The first thing I (*e20:*) is that they (*e21:*) writing this column.

However, most existing systems only make use of rather local features of these events, which cannot represent the prior knowledge humans have about these events and their "typical" order. As a result, existing systems almost always attempt to solve the situations shown in Example 12, even when they are actually presented with input as in Example 13. In this section, we propose such a resource in the form of a probabilistic knowledge base, constructed from a large New York Times (NYT) corpus. We hereafter name our resource TempRel *PRObabilistic knowledge Base*

(TemProb), which can potentially benefit many time-aware tasks. A few example entries of TemProb are shown in Table 4.6.

Table 4.6: **TemProb is a unique source of information of the temporal order that events *usually* follow.** The probabilities below do not add up to 100% because less frequent relations are omitted. The word sense numbers are not shown here for convenience.

Example Pairs		Before (%)	After (%)
accept	determine	42	26
ask	help	86	9
attend	schedule	1	82
accept	propose	10	77
die	explode	14	83
...			

Example 13: The original sentences in Example 12.
More than 10 people have (*e2:died*), police said. A car (*e1:exploded*) on Friday in the middle of a group of men playing volleyball.
The first thing I (*e20:ask*) is that they (*e21:help*) writing this column.

4.3.1 Related Work

The TempRel extraction task has a strong dependency on prior knowledge, as shown in our earlier examples. However, very limited attention has been paid to generating such a resource and to make use of it; to our knowledge, the TemProb proposed in this work is completely new. We find that the *time-sensitive relations* proposed in [119] is a close one in literature (although it is still very different). Jiang et al. [119] worked on the knowledge graph completion task. Based on YAGO2 [120] and Freebase [121], it manually selects a small number of relations that are time-sensitive (10 relations from YAGO2 and 87 relations from Freebase, respectively). Exemplar relations are wasBornIn$\rightarrow$diedIn$\rightarrow$ and graduateFrom$\rightarrow$workAt, where $\rightarrow$ means temporally before.

What we are trying to address in this section significantly differs from the time-sensitive relations in [119] in the following aspects. First, scale difference: [119] can only extract a small number of relations (<100), but we work on general semantic frames (tens of thousands) and the relations between any two of them, which we think has broader applications. Second, granularity difference: the smallest granularity in [119] is one year,[3] i.e., only when two events happened in different years can they know the temporal order of them, but this work can handle implicit temporal orders without having to refer to the physical time points of events (i.e., the granularity can be arbitrarily small). Third, domain difference: while [119] extracts time-sensitive relations from structured knowledge bases (where events are explicitly anchored to a time point), we extract relations from unstructured natural language text (where the physical time points may not even exist in text). Our task is more general and it allows us to extract much more relations, as reflected by the first difference above.

Another related work is the VerbOcean [122], which extracts TEMPRELs between pairs of verbs using manually designed lexico-syntactic patterns (there are in total 12 such patterns), in contrast to the automatic extraction method proposed in this work. In addition, the only temporal relation considered in VerbOceans is *before*, while we also consider relations such as *after*, *includes*, *included*, *simultaneous*, and *vague*. As expected, the total numbers of verbs and *before* relations in VerbOcean are about 3K and 4K, respectively, both of which are much smaller than TEMPROB, which contains 51K verb frames (i.e., disambiguated verbs), 9.2M $(verb1, verb2, relation)$ entries, and up to 80M TEMPRELs altogether.

All these differences necessitate the construction of a new resource for TEMPREL extraction, which we explain below.

4.3.2 TEMPROB: A Probabilistic Resource for TEMPRELs

In the TEMPREL extraction task, people have usually assumed that events are already given. However, to construct the desired resource, we need to extract events

[3]We notice that the smallest granularity in Freebase itself is one day, but [119] only used years.

(Sec. 4.3.2.1) and extract TempRels (Sec. 4.3.2.2), from a large, unannotated[4] corpus (Sec. 4.3.2.3). First, we considered semantic-frame based events, which could be detected via off-the-shelf semantic role labeling (SRL) tools. Then we applied a TempRel extractor on top of the events we extracted. We performed this procedure on more than 1 million NYT articles, spanning 20 years (1987-2007).[5] Finally, we also show some interesting statistics discovered in TemProb that may be useful for other tasks as well (Sec. 4.3.2.4).

4.3.2.1 Event Extraction

As we have introduced in Sec. 2.1, extracting events and the relations between them (e.g., coreference, causality, entailment, and temporal) has long been an active area in the NLP community. Generally speaking, an event is considered to be an action associated with corresponding participants involved in this action. In this work, following [123, 31, 32, 33], we consider semantic-frame based events, which can be directly detected via off-the-shelf semantic role labeling (SRL) tools. Specifically, we only look at verb semantic frames in this work due to the difficulty of getting TempRel annotation for nominal events.

When building a knowledge graph of TempRels, a certain level of abstraction is often preferred to be able to generalize. For example, given two events, "Jack is arrested because of robbery" and "John is arrested because of robbery", one question to ask is "are they the same or different?". One may think that they are different due to their difference between arguments (i.e., "Jack" vs. "John"), but an obvious disadvantage is that there are too many entities of different surface forms to account for in a limited dataset; more importantly, "rob" leading to "being arrested" is likely to be a common pattern in which their subjects play a minor role. Based on this intuition, we decide to start from the assumption that two events are considered to be in the same category as long as they share the same predicate (in other words, the knowledge base is built upon disambiguated predicates). As we show later, this

[4]Unannotated with TempRels.

[5]https://catalog.ldc.upenn.edu/LDC2008T19

assumption works reasonably well. We are aware that this level of abstraction may not be perfect, and future work can either perform clustering on those predicates to achieve a higher level of abstraction or plug in entity typing to achieve a finer level of abstraction.

4.3.2.2 TempRel Extraction

Given the events extracted in a given article, we next explain how the TempRels are extracted using a modified version of our system in Sec. 4.4.1. The TimeBank-Dense dataset [93] is known to have the best quality in terms of its high density of TempRels and is a benchmark dataset for the TempRel extraction task. Due to the slight event annotation difference in TimeBank-Dense, we collect our training data as follows. We first extracted all the verb semantic frames from the raw text of TimeBank-Dense using the verb SRL module from the Illinois Curator package [35, 36]. Then we only kept those semantic frames that were matched to an event in TimeBank-Dense (about 85% semantic frames were kept in this stage). By doing so, we could simply use the TempRel annotations provided in TimeBank-Dense. Hereafter the TimeBank-Dense dataset used in this section refers to this version unless otherwise specified.

We grouped the TempRels by the sentence distance of the two events of each relation.[6] Then we used the averaged perceptron algorithm [114] implemented in the Illinois LBJava package [116] to learn from the training data described above. Since only relations that have sentence distance 0 or 1 are annotated in TimeBank-Dense, we had two classifiers, one for same-sentence relations, and one for neighboring-sentence relations, respectively.

When generating TemProb, we need to process a large number of articles, so we adopted the greedy inference strategy described earlier due to its computational efficiency [95, 48]. Specifically, we applied the same-sentence relation classifier before the neighboring-sentence relation classifier; whenever a new relation is added in this

[6] That is, the difference of the appearance order of the sentence(s) containing the two target events.

article, a transitive graph closure is performed immediately. Doing so ensures that, if an edge is already labeled during the closure phase, it will not be labeled again, so conflicts are avoided.

4.3.2.3 Corpus

The corpus that we used to construct TemProb was comprised of NYT articles from 20 years (1987-2007).[7] It contains more than 1 million documents and we extracted events and corresponding features from each document using the Illinois Curator package [36] using the Amazon Web Services (AWS) cloud. In total, we discovered 51K unique verb semantic frames and 80M relations among them in the NYT corpus (15K of the verb frames had more than 20 relations extracted and 9K had more than 100 relations).

4.3.2.4 Useful Statistics in TemProb

We denote the set of all verb semantic frames by V. Let $D_i, i = 1, \ldots, N$ be the i-th document in our corpus, where N is the total number of documents. Let $G_i = (V_i, E_i)$ be the temporal graph inferred from D_i using the approach described above, where $V_i \subseteq V$ is the set of verbs/events extracted in D_i and $E_i = \{(v_m, v_n, r_{mn})\}_{m<n} \subseteq V_i \times V_i \times R$ is the edge set of D_i, which is composed of TempRel triplets; specifically, a TempRel triplet $(v_m, v_n, r_{mn}) \in E_i$ represents that in document D_i, the TempRel between v_m and v_n is r_{mn}. Due to the symmetry in TempRels, we only keep the triplets with $m < n$ in E_i. Assuming that the verbs in V_i are ordered by their appearance order in text, then $m < n$ means that in the i-th document, v_m appears earlier in text than v_n does.

Given the usual confusion between that one event is *temporally before* another and that one event is *physically appearing* before another in text, we will refer to temporally before as **T-Before** and physically before as **P-Before**. Using this language, for example, E_i only keeps the triplets that v_m is P-Before v_n in D_i.

[7] https://catalog.ldc.upenn.edu/LDC2008T19

We first show extreme cases that some events are *almost always* labeled as T-Before or T-After in the corpus. Specifically, for each pair of verbs $v_i, v_j \in V$, we define the following ratios:

$$\eta_b = \frac{C(v_i, v_j, before)}{C(v_i, v_j, before) + C(v_i, v_j, after)}, \eta_a = 1 - \eta_b, \tag{4.3}$$

where $C(v_i, v_j, r)$ is the count of v_i P-Before v_j with TEMPREL $r \in R$:

$$C(v_i, v_j, r) = \sum_{i=1}^{N} \sum_{(v_m, v_n, r_{mn}) \in E_i} \mathcal{I}_{\{v_m = v_i \& v_n = v_j \& r_{mn} = r\}}, \tag{4.4}$$

where $\mathcal{I}_{\{.\}}$ is the indicator function.

In Table 4.7, we show some event pairs with either $\eta_b > 0.9$ (upper part) or $\eta_a > 0.9$ (lower part). The temporal order of the pairs we show in Table 4.7 are almost deterministic, i.e., either T-Before or T-After with probability larger than 90%. We understand the remaining 10% (i.e. those #T-After's in the upper part and #T-Before's in the lower part) from two aspects: 1) system imperfection (recall that each G_i is of relatively low quality), and 2) complications brought by the difference in frame arguments (e.g., *"Jack is arrested"* is definitely possible to be T-After *"John is charged"*). Note that only pairs of η_b or $\eta_a > 0.9$ are shown in Table 4.7. Another usage of TEMPROB is that η_b and η_a can serve as a soft-decision and be incorporated in subsequent systems, which is exactly the kind of prior knowledge that we have expected.

In addition to the extreme cases shown in Table 4.7, we also show analysis of the distribution of preceding and following events in this section. For each verb v, we define the marginal count of v being P-Before to arbitrary verbs with TEMPREL $r \in R$ as $C(v, r) = \sum_{v_i \in V} C(v, v_i, r)$. Then for every other verb v', we define

$$P(v \text{ T-Before } v' | v \text{ T-Before}) \triangleq \frac{C(v, v', before)}{C(v, before)}, \tag{4.5}$$

which is the probability of v T-Before v', conditioned on v T-Before anything. Sim-

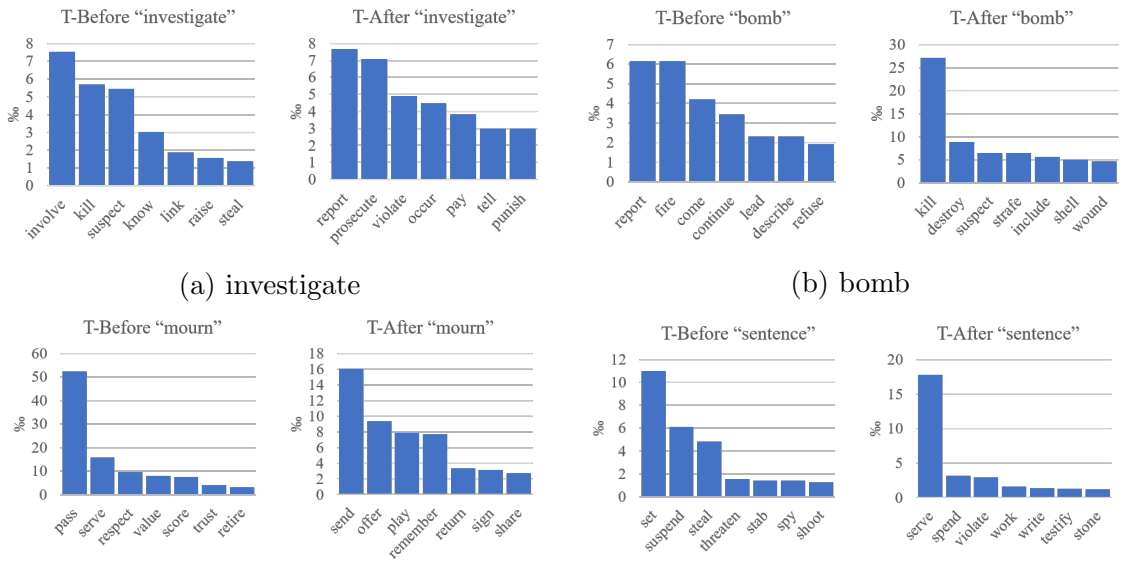
Table 4.7: Several extreme cases from TemProb, where some event is almost always labeled to be T-Before or T-After throughout the NYT corpus. By "extreme", we mean that either the probability of T-Before or T-After is larger than 90%. The upper part of the table shows the pairs that are both P-Before and T-Before, while the lower part shows the pairs that are P-Before but T-After. In TemProb, there are about 7K event pairs being extreme cases.

Example Pairs		#T-Before	#T-After
chop.01	taste.01	133	8
concern.01	protect.01	110	10
conspire.01	kill.01	113	6
debate.01	vote.01	48	5
dedicate.01	promote.02	67	7
fight.01	overthrow.01	98	8
achieve.01	desire.01	7	104
admire.01	respect.01	7	121
clean.02	contaminate.01	3	82
defend.01	accuse.01	13	160
die.01	crash.01	8	223
overthrow.01	elect.01	3	100

ilarly, we define

$$P(v \text{ T-After } v' | v \text{ T-After}) \triangleq \frac{C(v, v', after)}{C(v, after)}. \tag{4.6}$$

For a specific verb, e.g., v=*investigate*, each verb $v' \in V$ is sorted by the two conditional probabilities above. Then the most probable verbs that temporally precede or follow v are shown in Fig. 4.7, where the y-axes are the corresponding conditional probabilities. We can see reasonable event sequences like {*involve*, *kill*, *suspect*, *steal*}→*investigate*→{*report*, *prosecute*, *pay*, *punish*}, which indicates the possibility of using TemProb for event sequence predictions or story cloze tasks. There are also suspicious pairs like *know* in the T-Before list of *investigate* (Fig. 4.7a), *report* in the T-Before list of *bomb* (Fig. 4.7b), and *play* in the T-After list of *mourn* (Fig. 4.7c). Since the arguments of these verb frames are not considered here, whether these few seemingly counter-intuitive pairs come from system error or from a special context



(a) investigate

(b) bomb

(c) mourn

(d) sentence

Figure 4.7: Top events that most frequently precede or follow “investigate”, “bomb”, “mourn”, or “sentence” in time, sorted by their conditional probabilities in ‰. Word senses have been disambiguated and the “bomb” and “sentence” here are their verb meanings. There are some possible errors (e.g., *report* is T-Before *bomb*) and some unclear pairs (e.g., *know* is T-Before *investigate* and *play* is T-After *mourn*), but overall the event sequences discovered here are reasonable.

needs further investigation.

Now we have seen some interesting examples when we aggregate information from TemProb. In the next section, we will show quantitative analyses of how TemProb can help the TempRel task in this thesis.

4.3.3 Experiments

In the above, we have explained the construction of TemProb and shown some interesting examples from it, which were meant to visualize its correctness. In this section, we first quantify the correctness of the prior obtained in TemProb, and then show that TemProb can be used to improve existing TempRel extraction

systems.

4.3.3.1 Quality Analysis of TemProb

In Table 4.7, we showed examples with either η_b or $\eta_a > 0.9$. We argued that they *seem* correct. Here we quantify the "correctness" of η_b and η_a based on TimeBank-Dense. Specifically, we collected all the gold T-Before and T-After pairs. Let $\tau \in [0.5, 1)$ be a constant threshold. Imagine a naive predictor such that, for each pair of events v_i and v_j, if $\eta_b > \tau$, it predicts that v_i is T-Before v_j; if $\eta_a > \tau$, it predicts that v_i is T-After v_j; otherwise, it predicts that v_i is T-Vague to v_j. We expect that a higher η_b (or η_a) represents a higher confidence for an instance to be labeled T-Before (or T-After).

Table 4.8: Validating η_b and η_a from TemProb based on the T-Before and T-After examples in TimeBank-Dense. Performances are decomposed into same sentence examples (Dist=0) and contiguous sentence examples (Dist=1). A larger threshold leads to a higher precision, so η_b and η_a indeed represent a notion of confidence.

Threshold τ	Dist=0		Dist=1	
	P	R	P	R
0.5	65.6	61.3	58.5	53.3
0.6	69.8	44.5	60.5	36.9
0.7	74.6	29.2	63.6	18.7
0.8	81.0	13.9	64.8	6.9
0.9	**82.9**	5.0	**76.9**	1.2

Table 4.8 shows the performance of this predictor, which meets our expectation and thus justifies the validity of TemProb. As we gradually increase the value of τ in Table 4.8, the precision increases at roughly the same pace as τ, which indicates that the values of η_b and η_a[8] from TemProb indeed represent the confidence level. The decrease in recall is also expected because more examples are labeled as T-Vague when τ is larger.

[8]Recall the definitions of η_b and η_a in Eq. (4.3).

To further justify the quality, we also used another dataset that is not in the TempRel domain. Instead, we downloaded the EventCausality dataset[9] [43]. For each causally related pair *e1* and *e2*, if EventCausality annotates that *e1* causes *e2*, we changed it to be T-Before; if EventCausality annotates that *e1* is caused by *e2*, we changed it to be T-after. Therefore, based on the assumption that the cause event is T-Before the result event, we converted the EventCausality dataset to be a TempRel dataset and it thus could also be used to evaluate the quality of TemProb. We adopted the same predictor used in Table 4.8 with $\tau = 0.5$ and in Table 4.9, we compared it with two baselines: (i) always predicting T-Before and (ii) always predicting T-After. First, the accuracy (66.2%) in Table 4.9 is rather consistent with its counterpart in Table 4.8, confirming the stability of statistics from TemProb. Second, by directly using the prior statistics η_b and η_a from TemProb, we can improve the precision of both labels with a significant margin relative to the two baselines (17.0% for "T-Before" and 15.9% for "T-After"). Overall, the accuracy was improved by 11.5%.

Table 4.9: Further justification of η_b and η_a from TemProb on the EventCausality dataset. The thresholding predictor from Table 4.8 with $\tau = 0.5$ is used here. Compared to always predicting the majority label (i.e., T-Before in this case), $\tau = 0.5$ significantly improved the performance for both labels, with the overall accuracy improved by 11.5%.

System	T-Before		T-After		Acc.
	P	R	P	R	
T-Before Only	54.7	100.0	0	0	54.7
T-After Only	0	0	45.3	100	45.3
$\tau = 0.5$	**71.7**	63.3	**61.2**	69.8	**66.2**

4.3.3.2 Improving TempRel Extraction

Note that TimeBank-Dense was originally split into Train (22 docs), Dev (5 docs), and Test (9 docs). In the analysis here, we combined Train and Dev and we performed

[9] `http://cogcomp.org/page/resource_view/27`

3-fold cross validation on the 27 documents (in total about 10K relations) to tune the parameters in any classifier.

The purpose of TemProb was to improve TempRel extraction. We show it from two perspectives: How effective the prior distributions obtained from TemProb are (i) as features in local methods and (ii) as regularization terms in global methods. The results below were evaluated on the test split of TimeBank-Dense [93].

We first test how well the prior distributions from TemProb can be used as features in improving local methods for TempRel extraction. In Table 4.10, we used the original feature set previously used in Sec. 4.2, and added the prior distribution obtained from TemProb on top of it. Specifically, we added η_b (see Eq. (4.3)) and $\{f_r\}_{r\in R}$, respectively, where $\{f_r\}_{r\in R}$ is the prior distributions of all labels, i.e.,

$$f_r(v_i, v_j) = \frac{C(v_i, v_j, r)}{\sum_{r'\in R} C(v_i, v_j, r')},\ r \in R. \tag{4.7}$$

Recall function C is defined in Eq. (4.4). All comparisons were decomposed to same-sentence relations (Dist=0) and neighboring-sentence relations (Dist=1) for a better understanding of the behavior. All classifiers were trained using the averaged perceptron algorithm [114] and tuned by 3-fold cross validation.

From Table 4.10, we can see that simply adding η_b into the feature set could improve the original system F_1 by 1.8% (Dist=0) and 3.0% (Dist=1). If we further add as features the full set of prior distributions $\{f_r\}_{r\in R}$, the improvement comes to 2.7% and 6.5%, respectively. Noticing that the feature is more helpful for Dist=1, we think that it is because distant pairs usually have less lexical dependency and thus need more prior information provided by our new feature. With Dist=0 and Dist=1 combined (numbers not shown in the table), the third line improved the "original" by 4.7% in F_1 and by 5.1% in the temporal awareness F-score (another metric used in the TempEval3 workshop).

As mentioned earlier in Sec. 4.3.1, many systems adopt a global inference method via integer linear programming (ILP) [74] to enforce transitivity constraints over an entire temporal graph [101, 50, 124, 92, 9]. In addition to the usage shown above, the prior distributions from TemProb can also be used to regularize the conventional

Table 4.10: Using prior distributions derived from TemProb as features in an example local method. Incorporating η_b to the original feature set already yields better performance. By using the full set of prior distributions, $\{f_r\}_{r\in R}$, the final system improves the original in almost all metrics, and the improvement is statistically significant with p<0.005 per the McNemar's test.

Feature Set	Dist=0			Dist=1		
	P	R	F_1	P	R	F_1
Original	44.5	57.1	50.0	49.0	36.9	42.1
$+\eta_b$	46.2	58.9	51.8	**55.3**	38.1	45.1
$+\{f_r\}_{r\in R}$	**46.9**	**60.1**	**52.7**	51.3	**46.2**	**48.6**

Note The performances here are consistently lower than those in Table 4.8 because in Table 4.8, only T-Before and T-After examples are considered, but here all labels are taken into account and the problem is more practical and harder.

ILP formulation. Specifically, in each document, let $\mathcal{I}_r(ij) \in \{0,1\}$ be the indicator function of relation r for event i and event j; let $x_r(ij) \in [0,1]$ be the corresponding soft-max score obtained from the local classifiers (depending on the sentence distance between i and j). Then the ILP objective for global inference is formulated as follows.

$$\hat{\mathcal{I}} = \underset{\mathcal{I}}{\text{argmax}} \sum_{ij\in\mathcal{E}} \sum_{r\in R} (x_r(ij) + \underline{\lambda f_r(ij)})\mathcal{I}_r(ij) \tag{4.8}$$

$$\text{s.t.} \quad \underset{\text{(uniqueness)}}{\Sigma_r \mathcal{I}_r(ij) = 1}, \ \underset{\text{(symmetry)}}{\mathcal{I}_r(ij) = \mathcal{I}_{\bar{r}}(ji)},$$

$$\underset{\text{(transitivity)}}{\mathcal{I}_{r_1}(ij) + \mathcal{I}_{r_2}(jk) - \Sigma_{m=1}^{M} \mathcal{I}_{r_3^m}(ik) \leq 1},$$

for all distinct events i, j, and k, where $\mathcal{E} = \{ij \mid \text{sentence dist}(i,j) \leq 1\}$, λ adjusts the regularization term and was heuristically set to 0.5 in this work, $\bar{r}$ is the reverse relation of r, and M is the number of possible relations for r_3 when r_1 and r_2 are true. Note our difference from the ILP in [9] is the underlined regularization term $f_r(ij)$ (which itself is defined in Eq. (4.7)) obtained from TemProb.

We present results on the test split of TimeBank-Dense in Table 4.11, which is an ablation study showing step-by-step improvements in two metrics. In addition to the straightforward precision, recall, and F_1 metric, we also compared the F_1 of the

Table 4.11: Regularizing global methods by the prior distribution derived from TemProb. The "+" means adding a component on top of its preceding line. F_{aware} is the temporal awareness F-score, another evaluation metric used in TempEval3. The baseline system is to use (unregularized) ILP on top of the original system in Table 4.10. System 3 is the proposed. Per the McNemar's test, System 3 is significantly better than System 1 with $p<0.0005$.

No.	System	P	R	F_1	F_{aware}
1	Baseline	48.1	44.4	46.2	42.5
2	+Feature: $\{f_r\}_{r\in R}$	50.6	52.0	51.3	49.1
3	+Regularization	**51.3**	**53.0**	**52.1**	**49.6**

temporal awareness metric used in TempEval3 [3]. The awareness metric performs graph reduction and closure before evaluation so as to better capture how useful a temporal graph is. Details of this metric can be found in Sec. 4.1.3.

Table 4.12: Label-wise performance improvement of System 3 over System 1 in Table 4.11. We can see that incorporating TemProb improves the recall of *before* and *after*, and improves the precision of all labels, with a slight drop in the recall of *vague*.

Label	P	R	F_1
before	+0.3	+15	+6
after	+4	+4	+4
equal	+11	0	+2
includes	+17	0	+0.2
included	+8	0	+2
vague	+3	-4	-1

In Table 4.11, the baseline applied global ILP inference with transitivity constraints. Technically, it is to solve Eq. (4.8) with $\lambda = 0$ (i.e., unregularized) on top of the original system in Table 4.10. Apart from some implementation details, this baseline is also the same as many existing global methods as [50, 92]. System 2, "+Feature: $\{f_r\}_{r\in R}$", is to add prior distributions as features when training the local classifiers. Technically, the scores $x_r(ij)$'s in Eq. (4.8) used by baseline were changed. We know from Table 4.10 that adding $\{f_r\}_{r\in R}$ made the local decisions better. Here

the performance of System 2 shows that this was also the case for the global decisions made via ILP: both precision and recall got improved, and F_1 and awareness were both improved by a large margin, with 5.1% in F_1 and 6.6% in awareness F_1. On top of this, System 3 uses Eq. (4.8) (with $\lambda = 0.5$) to add regularizations to the conventional ILP formulation. The sum of these regularization terms represents a confidence score of how coherent the predicted temporal graph is to our TemProb, which we also want to maximize. Even though a considerable amount of information from TemProb had already been encoded as features (as shown by the large improvements by System 2), these regularizations were still able to further improve the precision, recall and awareness scores. To sum up, the total improvement over the baseline system brought by TemProb is 5.9% in F_1 and 7.1% in awareness F_1, both with a notable margin. Table 4.12 furthermore decomposes this improvement into each TempRel label.

To compare with state-of-the-art systems, which all used gold event properties (i.e., Tense, Aspect, Modality, and Polarity), we retrained System 3 in Table 4.11 with these gold properties and show the results in Table 4.13. We reproduced the results of CAEVO[10] [95] and Ning et al. (2017) [9][11] and evaluated them on the partial TimeBank-Dense test split.[12] Under both metrics, the proposed system achieved the best performance. An interesting fact is that even without these gold properties, our System 3 in Table 4.11 was already better than CAEVO (on Line 1) and [9] (on Line 2) in both metrics. This is appealing because in practice, those gold properties may not exist, but our proposed system can still generate state-of-the-art performance without them.

For readers who are interested in the complete TimeBank-Dense dataset, we also performed a naive augmentation as follows. Recall that System 3 only makes predictions to a subset of the complete TimeBank-Dense dataset. We kept this subset of predictions, and filled the missing predictions by [9]. Performances of this naively augmented proposed system are compared with CAEVO and [9] on the *complete*

[10] `https://github.com/nchambers/caevo`

[11] `http://cogcomp.org/page/publication_view/822`

[12] There are 731 relations in the partial TimeBank-Dense test split (201 *before*, 138 *after*, 39 *includes*, 31 *included*, 14 *simultaneous*, and 308 *vague*).

Table 4.13: Comparison of the proposed TempRel extraction method with two best-so-far systems using two metrics. Per the McNemar's test, Line 3 is better than Line 2 with $p<0.0005$.

No.	System	P	R	F_1	F_{aware}
Partial TimeBank-Dense: Focus of this work.*					
1	CAEVO	**52.3**	43.7	47.6	46.7
2	Ning et al. (2017) [9]	47.4	56.3	51.5	49.1
3	Proposed	50.0	**62.4**	**55.5**	**52.8**
Complete TimeBank-Dense: Naive augmentation.					
4	CAEVO	**51.8**	32.6	40.0	45.7
5	Ning et al. (2017) [9]	46.2	40.6	43.2	48.5
6	Proposed**	47.2	**42.4**	**44.7**	**49.2**

*Note that TempProb is only available for events extracted by SRL (See Sec. 4.3.2 for details).
**Augment the output of Line 3 with predictions from [9].

TimeBank-Dense dataset. We can see that by replacing with predictions from our proposed system, [9] got a better precision, recall, F_1, and awareness F_1, which is the new state-of-the-art on all reported performances on this dataset. Note that the awareness F_1 scores on Lines 4-5 are consistent with reported values in [9]. To our knowledge, Table 4.13 is the first in the literature to report performances in both metrics, and it is promising to see that the proposed method outperformed state-of-the-art methods in both metrics.

Up to now, we have argued that TempRel extraction is challenging partly due to its strong dependence on prior knowledge, and a resource of the temporal order that events *usually* follow is helpful. To construct such a resource, we automatically processed a large corpus from NYT with more than 1 million documents using a modified version of our TempRel extraction system in Sec. 4.2 and obtained TemProb. TemProb is a good showcase of the capability of such prior knowledge, and it has shown its power in improving existing TempRel extraction systems on a benchmark dataset, TimeBank-Dense. Similar to Sec. 4.2, TemProb can also be viewed as exploiting the structure of time. The difference is that in Sec. 4.2, we exploited transitivity structures as hard constraints, while in Sec. 4.3, we exploited

probabilistic structures as soft constraints/regularizations.

4.4 Multi-Axis Temporal Structure

As a continuation of the previous two sections, we propose another inherent structure in time which leads to a new dataset with improved annotation quality in this section. When introducing our structured learning approach in Sec. 4.2, we list some of the TempRel datasets available in this field in Table 4.3. These datasets were annotated by experts, but still suffered from low inter-annotator agreements (IAA). For instance, the IAAs of TimeBank-Dense, RED [125] and THYME-TimeML [126] were only below or near 60% (given that events are already annotated). Since a low IAA usually indicates that the task is difficult even for humans (see Examples 14-16), the community has been looking into ways to simplify the task, by reducing the label set, and by breaking up the overall, complex task into subtasks (e.g., getting agreement on which event pairs should have a relation, and then what that relation should be) [47, 125]. In contrast to other existing datasets, [91] achieved an agreement as high as 90%, but the scope of its annotation was narrowed down to a very special verb-clause structure.

(*e22*, *e23*), (*e24*, *e25*), and (*e26*, *e27*): TempRels that are difficult even for humans. Note that only relevant events are highlighted here.
Example 14: Serbian police tried to eliminate the pro-independence Kosovo Liberation Army and (*e22:restore*) order. At least 51 people were (*e23:killed*) in clashes between Serb police and ethnic Albanians in the troubled region.
Example 15: Service industries (*e24:showed*) solid job gains, as did manufacturers, two areas expected to be hardest (*e25:hit*) when the effects of the Asian crisis hit the American economy.
Example 16: We will act again if we have evidence he is (*e26:rebuilding*) his weapons of mass destruction capabilities, senior officials say. In a bit of television diplomacy, Iraq's deputy foreign minister (*e27:responded*) from Baghdad in less than one hour, saying that ...

Here we propose a new approach to handling these issues in TempRel annotation. In Sec. 4.4.1 and Sec. 4.4.2, we introduce multi-axis modeling to represent the temporal structure of events, based on which we can anchor events to different semantic axes; only events from the same axis will then be temporally compared (Sec. 4.2). As explained later, those event pairs in Examples 14-16 are difficult because they represent different semantic phenomena and belong to different axes. In addition, while we have represented an event pair using two time intervals (see Sec. 4.1.2 and Fig. 4.3), say, $[t^1_{start}, t^1_{end}]$ and $[t^2_{start}, t^2_{end}]$, we find that comparisons involving endpoints (e.g., t^1_{end} vs. t^2_{end}) are typically more difficult than comparing start-points (i.e., t^1_{start} vs. t^2_{start}); we attribute this to the ambiguity of expressing and perceiving durations of events [127]. As a result, we propose in Sec. 4.4.3 that TempRel annotation should focus on start-points. Using the proposed annotation scheme, a pilot study done by experts achieved a high IAA of 0.84 (Cohen's kappa) on a subset of TimeBank-Dense, in contrast to conventional IAAs in the 60's.

In addition to the low IAA issue, TempRel annotation is also known to be labor intensive. To address it, we use crowdsourcing to collect a new, high quality TempRel dataset, for the first time on this topic. This section will explain how the crowdsourcing quality was controlled and how *vague* relations were handled in Sec. 4.4.4, and present some statistics and the quality of the new dataset in Sec. 4.4.5. A baseline system is also shown to achieve much better performance on the new dataset, when compared with system performance in the literature (Sec. 4.4.6).

4.4.1 Temporal Structure of Events

Given a set of events, one important question in designing the TempRel annotation task is: Which pairs of events should have a relation? The answer depends on the modeling of the overall temporal structure of events.

TimeBank [89] laid the foundation for many subsequent TempRel corpora, e.g., Verb-Clause [91], TempEval3 [3], and TimeBank-Dense [95]. In TimeBank, the annotators were allowed to label TempRels between any pairs of events. This setup models the overall structure of events using a general graph, which made annotators

inadvertently overlook some pairs, resulting in low IAAs and many false negatives.

Example 17: Dense Annotation Scheme.
Serbian police (*e28:tried*) to (*e29:eliminate*) the pro-independence Kosovo Liberation Army and (*e22:restore*) order. At least 51 people were (*e23:killed*) in clashes between Serb police and ethnic Albanians in the troubled region.
Given 4 Non-Generic events above, the dense scheme presents 6 pairs to annotators one by one: (*e28*, *e29*), (*e28*, *e22*), (*e28*, *e23*), (*e29*, *e22*), (*e29*, *e23*), and (*e22*, *e23*). Apparently, not all pairs are well-defined, e.g., (*e29*, *e23*) and (*e22*, *e23*), but annotators are forced to label all of them.

To address this issue, [93] proposed a dense annotation scheme, TimeBank-Dense, which annotates all event pairs within a sliding, two-sentence window (see Example 17). It requires all TEMPRELS between GENERIC[13] and NON-GENERIC events to be labeled as *vague*, which in our language here, models the overall temporal structure by *two disjoint time-axes*: one for the NON-GENERIC and the other one for the GENERIC.

However, as shown by Examples 14-16 in which the highlighted events are all NON-GENERIC, the TEMPRELS may still be ill-defined: In Example 14, Serbian police tried to restore order but ended up with conflicts. It is reasonable to argue that the attempt to *e22:restore* order happened *before* the conflict where 51 people were *e23:killed*; or, 51 people had been *killed* but order had not been *restored* yet, so *e22:restore* is *after e23:killed*. Similarly, in Example 15, service industries and manufacturers were originally expected to be hardest *e25:hit* but actually *e24:showed* gains, so *e25:hit* is *before e24:showed*; however, one can also argue that the two areas had *showed* gains but had not been *hit*, so *e25:hit* is *after e24:showed*. Again, *e26:rebuilding* is a hypothetical event: "we will act if *rebuilding* is true". Readers do not know for sure if "he is already rebuilding weapons but we have no evidence", or "he will be building weapons in the future", so annotators may disagree on the relation between *e26:rebuilding* and *e27:responded*. As another way to resolve confu-

[13]For example, *lions eat meat* is GENERIC.

Table 4.14: The interpretation of various event types that are not on the main axis in the proposed multi-axis modeling. The names are rather straightforward; see examples for each in Example 18.

Event Type	Category
Intention, Opinion	On an orthogonal axis
Hypothesis, Generic	On a parallel axis
Negation	Not on any axis
Static, Recurrent	Other

sions, TimeBank-Dense resorted to a 80% confidence rule: annotators were allowed to choose a label if one is 80% sure that it was the writer's intent. However, as pointed out by TimeBank-Dense, annotators are likely to have rather different understandings of 80% confidence and it will still end up with disagreements. Despite minimizing missing annotations, the dense scheme forces annotators to label many such ill-defined pairs, resulting in low IAA.

In contrast to these annotation difficulties, humans can easily grasp the meaning of news articles, implying a potential gap between the difficulty of the annotation task and the one of understanding the actual meaning of the text. In Examples 14-16, the writers did not intend to explain the TempRels between those pairs, and the original annotators of TimeBank[14] did not label relations between those pairs either, which indicates that both writers and readers did not think the TempRels between these pairs were crucial. Instead, what is crucial in these examples is that "Serbian police *tried* to restore order but *killed* 51 people", that "two areas were *expected* to be hit but *showed* gains", and that "*if* he rebuilds weapons *then* we will act." To "*restore* order", to be "hardest *hit*", and "if he was *rebuilding*" were only the intention of police, the opinion of economists, and the condition to *act*, respectively, and whether or not they actually happen is not the focus of those writers.

This discussion suggests that in TimeBank-Dense, a single axis is too restrictive to represent the complex structure of Non-Generic events. Instead, we need a temporal structure which is more restrictive than a general graph (as in TimeBank)

[14]Recall that they were given the entire article and only salient relations would be annotated.

so that annotators can focus on relation annotation (rather than looking for pairs first), but also more flexible than a single axis so that ill-defined relations are not forcibly annotated. Specifically, we need axes for intentions, opinions, hypotheses, etc., in addition to the main axis of an article. We propose *multi-axis modeling*, as defined in Table 4.14. Following the proposed modeling, Examples 14-16 can be represented as in Fig. 4.8. This modeling aims at capturing what the author has explicitly expressed and it only asks annotators to look at comparable pairs, rather than forcing them to make decisions on often vaguely defined pairs.

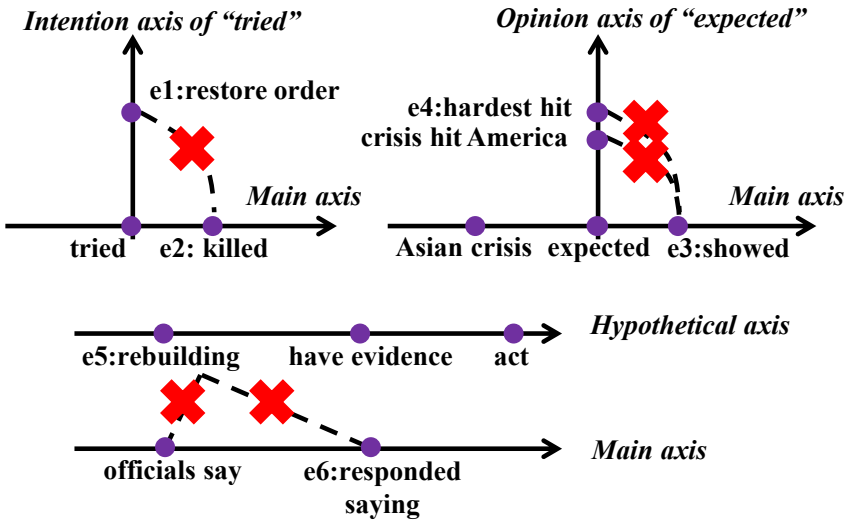


Figure 4.8: A multi-axis view of Examples 14-16. Only events on the same axis are compared.

The names of those categories in Table 4.14 are straightforward. Here we further provide examples for each of them in Example 18. Note that most of them are consistent with the definitions in the literature, with one exception for INTENTION. In TimeML [128], there are two types of intentions, I-Action (e.g., *attempt, try* and *promise*) and I-State (e.g., *believe, intend* and *want*). But our definition of intention is the actual intent of these verbs. For example, in Example 18, *e30* and *e31* are INTENTION. This definition is more general so that verbs that are not I-Action or I-State can still create orthogonal axis of intention, e.g., the verb "allocated" in the

sentence of *e31*.

Example 18
[Orthogonal axis] Intention/Opinion I plan/want to (*e30:leave*) tomorrow. The mayor has allocated funds to (*e31:build*) a museum. I think he will (*e32:win*) the race.
[Parallel axis] Hypothesis/Generic If I'm (*e33:elected*), I will cut income tax. If I'm elected, I will (*e34:cut*) income tax. Fruit (*e35:contains*) water. Lions (*e36:hunt*) zebras.
[Not on any axis] Negation The financial assistance from the Wolrd Bank is not (*e37:helping*). They don't (*e38:want*) to play with us. He failed to (*e39:find*) buyers.
[Other] Static/Recurrent He (*e40:is*) brave. New York (*e41:is*) on the east coast. The shuttle will be (*e42:departing*) at 6:30am every day.

In practice, we annotate one axis at a time: we first classify if an event is anchorable onto a given axis (this is also called the anchorability annotation step); then we annotate every pair of anchorable events (i.e., the relation annotation step); finally, we can move to another axis and repeat the two steps above. Note that ruling out cross-axis relations is only a strategy we adopt in this paper to separate well-defined relations from ill-defined relations. We are not claiming that cross-axis relations are unimportant; instead, as shown in Fig. 4.9, we think that cross-axis relations are a different semantic phenomenon that requires additional investigation.

4.4.2 Comparisons with Existing Work

There have been other proposals of temporal structure modelings [101, 129], but in general, the semantic phenomena handled in this work are very different and complementary to them. Bramsen et al. (2006) [101] introduces "temporal segments" (a

fragment of text that does not exhibit abrupt changes) in the medical domain. Similarly, their temporal segments can also be considered as a special temporal structure modeling. But a key difference is that [101] only annotates inter-segment relations, ignoring intra-segment ones. Since those segments are usually large chunks of text, the granularity of the semantics handled in [101] is very coarse (as pointed out by [101]) and is thus different from that in this thesis work.

Bethard et al. (2012) [129] propose a tree structure for children's stories, which "typically have simpler temporal structures", as they pointed out. Moreover, in their annotation, an event can only be linked to a single nearby event, even if multiple nearby events may exist, whereas we do not have such restrictions.

In addition, some of the semantic phenomena in Table 4.14 have been discussed in existing work. Here we compare with them for a better positioning of the proposed scheme.

4.4.2.1 Axis Projection

TimeBank-Dense handled the incomparability between main-axis events and HYPOTHESIS/NEGATION by *treating an event as having occurred* if the event is HYPOTHESIS/NEGATION.[15] In our multi-axis modeling, the strategy adopted by TimeBank-Dense falls into a more general approach, "axis projection". That is, projecting events across different axes to handle the incomparability between any two axes (not limited to HYPOTHESIS/NEGATION). Axis projection works well for certain event pairs like *Asian crisis* and *e25:hardest hit* in Example 15: as in Fig. 4.8, *Asian crisis* is *before expected*, which is again *before e25:hardest hit*, so *Asian crisis* is *before e25:hardest hit*.

Generally, however, since there is no direct evidence that can guide the projection, annotators may have different projections (imagine projecting *e26:rebuilding* onto the main axis: Is it in the past or in the future?). As a result, axis projection requires many specially designed guidelines or strong external knowledge. Annotators have

[15] In the case of Example 16, it is to treat *rebuilding* as actually happened and then link it to *responded*.

to rigidly follow the sometimes counter-intuitive guidelines or "guess" a label instead of looking for evidence in the text.

When strong external knowledge is involved in axis projection, it becomes a reasoning process and the resulting relations are a different type. For example, a reader may reason that in Example 16, it is well-known that they did "act again", implying his *e26:rebuilding* had happened and is *before e27:responded*. Another example is in Fig. 4.9. It is obvious that relations based on these projections are not the same and are more challenging than those same-axis relations, so in the current stage, we should focus on same-axis relations only.

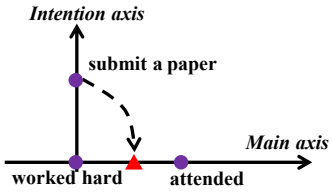


Figure 4.9: In *I worked hard to submit a paper . . . I attended the conference*, the projection of *submit a paper* onto the main axis is clearly *before attended*. However, this projection requires strong external knowledge that a paper should be submitted before attending a conference. Again, this projection is only a guess based on our external knowledge and it is still open whether the paper is submitted or not.

4.4.2.2 Introduction of the Orthogonal Axes

Another prominent difference from earlier work is the introduction of orthogonal axes, which has not been used in any existing work we know of. A special property is that the intersection event of two axes can be compared to events from both, which can sometimes bridge events; e.g., in Fig. 4.8, *Asian crisis* is seemingly *before hardest hit* due to their connections to *expected*. Since *Asian crisis* is on the main axis, it seems that *e25:hardest hit* is on the main axis as well. However, the "*hardest hit*" in "*Asian crisis before hardest hit*" is only a projection of the original *e25:hardest hit* onto the real axis and is valid only when this OPINION is true.

Nevertheless, OPINIONS are not always true and INTENTIONS are not always fulfilled. In Example 19, *e43:sponsoring* and *e44:resolve* are the opinions of the West and the speaker, respectively; whether or not they are true depends on the authors' implications or the readers' understandings, which is often beyond the scope of TEMPREL annotation.[16] Example 20 demonstrates a similar situation for INTENTIONS: when reading the sentence of *e45:report*, people are inclined to believe that it is fulfilled. But if we read the sentence of *e46:report*, we have reason to believe that it is not. When it comes to *e47:tell*, it is unclear if everyone told the truth. The existence of such examples indicates that orthogonal axes are a better modeling for INTENTIONS and OPINIONS.

Example 19: Opinion events may not always be true.
He is ostracized by the West for (*e43:sponsoring*) terrorism.
We need to (*e44:resolve*) the deep-seated causes that have resulted in these problems.
Example 20: Intentions may not always be fulfilled.
A passerby called the police to (*e45:report*) the body.
A passerby called the police to (*e46:report*) the body. Unfortunately, the line was busy.
I asked everyone to (*e47:tell*) the truth.

4.4.2.3 Differences from Factuality

Event modality has been discussed in many existing event annotation schemes, e.g., Event Nugget [14], Rich ERE [15], and RED. Generally, an event is classified as *Actual* or *Non-Actual*, a.k.a. factuality [130, 131].

The main-axis events defined in this chapter seem to be very similar to *Actual* events, but with several important differences: First, future events are *Non-Actual* because they indeed have not happened, but they may be on the main axis. Second, events that are not on the main axis can also be *Actual* events, e.g., intentions that are fulfilled, or opinions that are true. Third, as demonstrated by Examples 19-20,

[16]For instance, there is undoubtedly a *causal* link between *e43:sponsoring* and *ostracized*.

identifying anchorability as defined in Table 4.14 is relatively easy, but judging if an event actually happened is often a high-level understanding task that requires an understanding of the entire document or external knowledge.

Below is a detailed analysis of the difference between *Anchorable* (onto the main axis) and *Actual* on a subset of RED [125]. We randomly selected 5 documents from RED, where there are 314 events, 166 of which are verbs (we only handle verb events). Along with another NLP researcher, we annotated the anchorability of these 166 verb events independently without looking at the original REALIS annotation from RED, and we achieved a Cohen's kappa of 0.88 in anchorability annotation, consistent with their Cohen's kappa achieved on MATRES. To aggregate the result from two experts, we marked an event as *Anchorable* only when both annotators labeled *Anchorable*. As for REALIS labeling in RED, we grouped GENERIC, HYPOTHETICAL, and HEDGED into a single label of *Non-Actual*.

Table 4.15: Comparison between anchorability and factuality on a subset of verb events randomly selected from RED.

		Anchorable	
		Yes	No
Actual	Yes	108	21
	No	0	37

The comparison between *Anchorable* and *Actual* is shown in Table 4.15. On this subset of 166 events, we did not see *Anchorable* events that are *Non-Actual* because such cases are indeed infrequent in practice; the only difference is that we annotated 21 events as *Non-Anchorable*, while RED annotated them as *Actual*. Among the 21 different cases, 11 are INTENTION, 4 are OPINION, and 6 are STATIC. Note in total RED labeled 4 *Actual* but NEGATION, and we have treated these 4 cases as *Non-actual* in Table 4.15. Typical examples from each category are shown in Example 21. Note that if we calculate the McNemar's statistics based on Table 4.15, *Anchorable* and *Actual* are statistically different with $p \ll 0.001$.

Example 21: Typical cases that RED annotated *Actual* and we annotated *Non-Anchorable*.
Libya has since agreed to (*e48:pay*) compensation to the families of the Berlin disco victims as well as the families of the victims of the 1988 Pan Am 103 bombing over Lockerbie, Scotland, which killed 270 people, including 189 Americans. [**We think it is Intention**]
Gadhafi had long been ostracized by the West for (*e49:sponsoring*) terrorism, but in recent years sought to emerge from his pariah status by abandoning weapons of mass destruction and renouncing terrorism in 2003. [**We think it is Opinion**]
We need to resolve the deep-seated causes that have resulted in these problems, Premier Wen said in an interview with Hong Kong-(*e50:based*) Phoenix Television. [**We think it is Static**]
Fuel prices had been frozen for six years, but the government said it could no longer afford to (*e51:subsidize*) them. [**We think it is Negation**]

4.4.3 Interval Splitting

All existing annotation schemes adopt the interval representation of events [90] and there are 13 relations between two intervals (for readers who are not familiar with it, please see Fig. 4.3 in the appendix). To reduce the burden of annotators, existing schemes often resort to a reduced set of the 13 relations. For instance, [1] merged all the overlap relations into a single relation, *overlap*. Bethard et al. (2007) [91], Do et al. (2012) [92], and O'Gorman et al. (2016) [125] all adopted this strategy. In [93], they further split *overlap* into *includes*, *included* and *equal*.

Let $[t^1_{start}, t^1_{end}]$ and $[t^2_{start}, t^2_{end}]$ be the time intervals of two events (with the implicit assumption that $t_{start} \leq t_{end}$). Instead of reducing the relations between two intervals, we try to explicitly compare the time points (see Fig. 4.10). In this way, the label set is simply *before, after* and *equal*,[17] while the expressivity remains the same. This interval splitting technique has also been used in [132].

In addition to same expressivity, interval splitting can provide even more information when the relation between two events is *vague*. In the conventional setting, imagine that the annotators find that the relation between two events can be either *before* or *before and overlap*. Then the resulting annotation will have to be *vague*,

[17]We will discuss *vague* in Sec. 4.4.4.

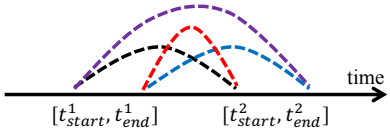


Figure 4.10: The comparison of two event time intervals, $[t^1_{start}, t^1_{end}]$ and $[t^2_{start}, t^2_{end}]$, can be decomposed into four comparisons t^1_{start} vs. t^2_{start}, t^1_{start} vs. t^2_{end}, t^1_{end} vs. t^2_{start}, and t^1_{end} vs. t^2_{end}, without loss of generality.

although the annotators actually agree on the relation between t^1_{start} and t^2_{start}. Using interval splitting, however, such information can be preserved.

An obvious disadvantage of interval splitting is the increased number of annotations needed (4 point comparisons vs. 1 interval comparison). In practice, however, it is usually much fewer than 4 comparisons. For example, when we see $t^1_{end} < t^2_{start}$ (as in Fig. 4.10), the other three can be skipped because they can all be inferred. Moreover, although the number of annotations is increased, the workload for human annotators may still be the same, because even in the conventional scheme, they still need to think of the relations between start- and end-points before they can make a decision.

During our pilot annotation, the annotation quality dropped significantly when the annotators needed to reason about relations involving end-points of events. Table 4.16 shows four metrics of task difficulty when only t^1_{start} vs. t^2_{start} or t^1_{end} vs. t^2_{end} are annotated. Non-anchorable events were removed for both jobs. The first two metrics, qualifying pass rate and survival rate, are related to the two quality control protocols (see Sec. 4.4.4.1 for details). We can see that when annotating the relations between end-points, only one out of ten crowdsourcers (11%) could successfully pass our qualifying test; and even if they had passed it, half of them (56%) would have been kicked out in the middle of the task. The third line is the overall accuracy on gold set from all crowdsourcers (excluding those who did not pass the qualifying test), which drops from 67% to 37% when annotating end-end relations. The last line is the average response time per annotation and we can see that it takes much longer to label an end-end TempRel (52s) than a start-start TempRel (33s). This important discovery indicates that the TempRels between end-points is probably

governed by a different linguistic phenomenon.

Table 4.16: Annotations involving the end-points of events are found to be much harder than only comparing the start-points.

Metric	t^1_{start} vs. t^2_{start}	t^1_{end} vs. t^2_{end}
Qualification pass rate	50%	11%
Survival rate	74%	56%
Accuracy on gold	67%	37%
Avg. response time	33s	52s

We hypothesize that the difficulty is a mixture of how durative events are expressed (by authors) and perceived (by readers) in natural language. In cognitive psychology, [127] discovered that human readers take longer to perceive durative events than punctual events, e.g., *owe 50 bucks* vs. *lost 50 bucks*. From the writer's standpoint, durations are usually fuzzy [133], or assumed to be a prior knowledge of readers (e.g., college takes 4 years and watching an NBA game takes a few hours), and thus not always written explicitly. Given all these reasons, we ignore the comparison of end-points in this work, although event duration is indeed another important task.

4.4.4 Annotation Scheme Design

To summarize, with the proposed multi-axis modeling (Sec. 4.4.1) and interval splitting (Sec. 4.4.3), our annotation scheme is two-step. First, we mark every event candidate as being temporally *Anchorable* or not (based on the time axis we are working on). Second, we adopt the dense annotation scheme to label TempRels only between *Anchorable* events. Note that we only work on verb events in this paper, so non-verb event candidates are also deleted in a preprocessing step. We design crowdsourcing tasks for both steps and as we show later, high crowdsourcing quality was achieved on both tasks. In this section, we will discuss some practical issues.

4.4.4.1 Quality Control for Crowdsourcing

We take advantage of the quality control feature in CrowdFlower in our crowdsourcing jobs. For any job, a set of examples are annotated by experts beforehand, which is considered gold and will serve two purposes. (i) Qualifying test: Any crowdsourcer who wants to work on this job has to pass with 70% accuracy on 10 questions randomly selected from the gold set. (ii) Surviving test: During the annotation process, questions from the gold set will be randomly given to crowdsourcers without notice, and one has to maintain 70% accuracy on the gold set till the end of the annotation; otherwise, he or she will be forbidden from working on this job anymore and all his/her annotations will be discarded. At least 5 different annotators are required for every judgement and by default, the majority vote will be the final decision.

4.4.4.2 Vague Relations

How to handle *vague* relations is another issue in temporal annotation. In non-dense schemes, annotators usually skip the annotation of a *vague* pair. In dense schemes, a majority agreement rule is applied as a postprocessing step to back off a decision to *vague* when annotators cannot pass a majority vote [93], which reminds us that annotators often label a *vague* relation as non-vague due to lack of thinking.

We decide to proactively reduce the possibility of such situations. As mentioned earlier, our label set for t^1_{start} vs. t^2_{start} is *before*, *after*, *simultaneous* and *vague*. We ask two questions: Q1=Is it possible that t^1_{start} is before t^2_{start}? Q2=Is it possible that t^2_{start} is before t^1_{start}? Let the answers be A1 and A2. Then we have a one-to-one mapping as follows: A1=A2=yes$\mapsto$*vague*, A1=A2=no$\mapsto$*simultaneous*, A1=yes, A2=no$\mapsto$*before*, and A1=no, A2=yes$\mapsto$*after*. An advantage is that one will be prompted to think about all possibilities, thus reducing the chance of overlook.

4.4.5 Corpus Statistics and Quality

In this section, we first focus on annotations on the main axis, which is usually the primary storyline and thus has most events. Before launching the crowdsourcing tasks, we checked the IAA between two experts on a subset of TimeBank-Dense (about 100 events and 400 relations). A Cohen's kappa of .85 was achieved in the first step: anchorability annotation. Only those events that both experts labeled *Anchorable* were kept before they moved onto the second step: relation annotation, for which the Cohen's kappa was .90 for Q1 and .87 for Q2. Table 4.17 furthermore shows the distribution, Cohen's kappa, and F_1 of each label. We can see the kappa and F_1 of *vague* (κ=.75, F_1=.81) are generally lower than those of the other labels, confirming that temporal *vagueness* is a more difficult semantic phenomenon. Nevertheless, the overall IAA shown in Table 4.17 is a significant improvement compared to existing datasets.

Table 4.17: IAA of two experts' annotations in a pilot study on the main axis. Notations: **b**efore, **a**fter, **e**qual, and **v**ague.

	b	a	e	v	Overall
Distribution	.49	.23	.02	.26	1
IAA: Cohen's κ	.90	.87	1	.75	.84
IAA: F_1	.92	.93	1	.81	.90

With the improved IAA confirmed by experts, we sequentially launched the two-step crowdsourcing tasks through CrowdFlower on top of the same 36 documents of TimeBank-Dense. To evaluate how well the crowdsourcers performed on our task, we calculate two quality metrics: accuracy on the gold set and the Worker Agreement with Aggregate (WAWA). WAWA indicates the average number of crowdsourcers' responses agreed with the aggregate answer (we used majority aggregation for each question). For example, if N individual responses were obtained in total, and n of them were correct when compared to the aggregate answer, then WAWA is simply n/N. In the first step, crowdsourcers labeled 28% of the events as *Non-Anchorable* to the main axis, with an accuracy on the gold of .86 and a WAWA of .79.

With *Non-Anchorable* events filtered, the relation annotation step was launched

as another crowdsourcing task. The label distribution is b=.50, a=.28, e=.03, and v=.19 (consistent with Table 4.17). In Table 4.18, we show the annotation quality of this step using accuracy on the gold set and WAWA. We can see that the crowdsourcers achieved a very good performance on the gold set, indicating that they are consistent with the authors who created the gold set; these crowdsourcers also achieved a high-level agreement under the WAWA metric, indicating that they are consistent among themselves. These two metrics indicate that the annotation task is now well-defined and easy to understand even by non-experts.

Table 4.18: Quality analysis of the relation annotation step of MATRES. "Q1" and "Q2" refer to the two questions crowdsourcers were asked (see Sec. 4.4.4.2 for details). Line 1 measures the level of consistency between crowdsourcers and the authors and line 2 measures the level of consistency among the crowdsourcers themselves.

No.	Metric	Q1	Q2	All
1	Accuracy on Gold	.89	.88	.88
2	WAWA	.82	.81	.81

We continued to annotate INTENTION and OPINION which create orthogonal branches on the main axis. In the first step, crowdsourcers achieved an accuracy on gold of .82 and a WAWA of .89. Since only 16% of the events are in this category and these axes are usually very short (e.g., *allocate funds to build a museum.*), the annotation task is relatively small and two experts took the second step and achieved an agreement of .86 (F_1).

We name our new dataset *MATRES* for Multi-Axis Temporal RElations for Start-points. Each individual judgement cost us \$0.01 and MATRES in total cost about \$400 for 36 documents.

To get another checkpoint of the quality of the new dataset, we compare with the annotations of TimeBank-Dense. TimeBank-Dense has 1.1K verb events, between which 3.4K event-event (EE) relations are annotated. In the new dataset, 72% of the events (0.8K) are anchored onto the main axis, resulting in 1.6K EE relations, and 16% (0.2K) are anchored onto orthogonal axes, resulting in 0.2K EE relations. The following comparison is based on the 1.8K EE relations in common. Moreover,

since TimeBank-Dense annotations are for intervals instead of start-points only, we converted TimeBank-Dense's interval relations to start-point relations (e.g., if A includes B, then t^A_{start} is before t^B_{start}).

Table 4.19: An evaluation of MATRES against TimeBank-Dense. Horizontal: MATRES. Vertical: TimeBank-Dense (with interval relations mapped to start-point relations). Please see explanation of these numbers in text.

	b	a	e	v	All
b	455	11	5	42	513
a	45	309	16	68	438
e	13	7	2	10	32
v	450	138	20	192	800
All	963	465	43	312	1783

The confusion matrix is shown in Table 4.19. A few remarks about how to understand it: First, when TimeBank-Dense labels *before* or *after*, MATRES also has a high-probability of having the same label (b=455/513=.89, a=309/438=.71); when MATRES labels *vague*, TimeBank-Dense is also very likely to label *vague* (v=192/312=.62). This indicates the *high agreement level* between the two datasets if the interval- or point-based annotation difference is ruled out. Second, many *vague* relations in TimeBank-Dense are labeled as *before*, *after* or *simultaneous* in MATRES. This is expected because TimeBank-Dense annotates relations between *intervals*, while MATRES annotates *start-points*. When durative events are involved, the problem usually becomes more difficult and interval-based annotation is more likely to label *vague* (see earlier discussions in Sec. 4.4.3). Example 22 shows three typical cases, where *e52:became*, *e55:backed*, *e56:rose* and *e57:extending* can be considered durative. If only their start-points are considered, the crowdsourcers were correct in labeling *e52* before *e53*, *e54* after *e55*, and *e56* equal to *e57*, although TimeBank-Dense says *vague* for all of them. Third, *simultaneous* seems to be the relation that the two datasets mostly disagree on, which is probably due to crowdsourcers' lack of understanding in time granularity and event coreference. Although *simultaneous* relations only constitute a small portion in all relations, they need further investigation.

Example 22: Typical cases that TimeBank-Dense annotated *vague* but MATRES annotated *before*, *after*, and *simultaneous*, respectively.
At one point , when it (*e52:became*) clear controllers could not contact the plane, someone (*e53:said*) a prayer. *TimeBank-Dense:* vague*; MATRES:* before
The US is bolstering its military presence in the gulf, as President Clinton (*e54:discussed*) the Iraq crisis with the one ally who has (*e55:backed*) his threat of force, British prime minister Tony Blair. *TimeBank-Dense:* vague*; MATRES:* after
Average hourly earnings of nonsupervisory employees (*e56:rose*) to $12.51. The gain left wages 3.8 percent higher than a year earlier, (*e57:extending*) a trend that has given back to workers some of the earning power they lost to inflation in the last decade. *TimeBank-Dense:* vague*; MATRES:* simultaneous

4.4.6 Experiments

We develop a baseline system for TempRel extraction on MATRES, assuming that all the events and axes are given. The following commonly-used features for each event pair are used: (i) The part-of-speech (POS) tags of each individual event and of its neighboring three words. (ii) The sentence and token distance between the two events. (iii) The appearance of any modal verb between the two event mentions in text (i.e., will, would, can, could, may and might). (iv) The appearance of any temporal connectives between the two event mentions (e.g., before, after and since). (v) Whether the two verbs have a common synonym from their synsets in WordNet [134]. (vi) Whether the input event mentions have a common derivational form derived from WordNet. (vii) The head words of the preposition phrases that cover each event, respectively. And (viii) event properties such as Aspect, Modality, and Polarity that come with the TimeBank dataset and are commonly used as features.

The proposed baseline system uses the averaged perceptron algorithm to classify the relation between each event pair into one of the four relation types. We adopted the same train/dev/test split of TimeBank-Dense, where there are 22 documents in train, 5 in dev, and 9 in test. Parameters were tuned on the train-set to maximize its

F_1 on the dev-set, after which the classifier was retrained on the union of train and dev. A detailed analysis of the baseline system is provided in Table 4.20. The performance on *simultaneous* and *vague* is lower than on *before* and *after*, probably due to shortage in these labels in the training data and the inherent difficulty in event coreference and temporal vagueness. We can see, though, that the overall performance on MATRES is much better than those in the literature for TempRel extraction, which used to be in the low 50's [95, 9]. The same system was also retrained and tested on the original annotations of TimeBank-Dense (Line "Original"), which confirms the significant improvement if the proposed annotation scheme is used. Note that we *do not* mean to say that the proposed baseline system itself is better than other existing algorithms, but rather that the proposed annotation scheme and the resulting dataset lead to better defined machine learning tasks. In the future, more data can be collected and used with advanced techniques such as ILP [92], structured learning [9] or multi-sieve [95].

Table 4.20: Performance of the proposed baseline system on MATRES. Line "Original" is the same system retrained on the original TimeBank-Dense and tested on the same subset of event pairs. Due to the limited number of *simultaneous* examples, the system did not make any *simultaneous* predictions on the testset.

	Training			Testing		
	P	R	F_1	P	R	F_1
Before	.74	.91	.82	.71	.80	.75
After	.73	.77	.75	.55	.64	.59
Equal	1	.05	.09	-	-	-
Vague	.75	.28	.41	.29	.13	.18
Overall	**.73**	**.81**	**.77**	**.66**	**.72**	**.69**
Original	.44	.67	.53	.40	.60	.48

4.5 CogCompTime: A Combination of Above

In summary, the contribution of this thesis work on TempRel extraction is exploiting multiple forms of structure of time in inference, learning, and data annotation.

Specifically, we have introduced how to make use of the transitivity structure in Sec. 4.2, the probabilistic structure as commonsense knowledge in Sec. 4.3, and the multi-axis structure in Sec. 4.4. In this section, we will show the effectiveness of these proposals if they are combined (the resulting system is called CogCompTime). We will first talk about incorporating all of them in a feature-based method, and then switch to a neural framework to achieve even further improvement. We will see that the state-of-the-art on TempRel extraction can be improved significantly by the proposals in this thesis.

4.5.1 Feature-based Method

As we described in earlier sections, TempRel extraction can be seen as a graph extraction problem, where the nodes represent events, and the edges represent TempRels. With all the nodes extracted, the TempRel component is to make predictions on the labels of those edges. Here, the label set for TempRels is *before*, *after*, *simultaneous*, and *vague*.

For each pair of nodes, the following features are used to predict the labels. (i) The part-of-speech (POS) tags from each individual verb and from its neighboring three words. (ii) The distance between them in terms of the number of tokens. (iii) The modal verbs between the event mention (i.e., will, would, can, could, may and might). (iv) The temporal connectives between the event mentions (e.g., before, after and since). (v) Whether the two verbs have a common synonym from their synsets in WordNet [134]. (vi) Whether the input event mentions have a common derivational form derived from WordNet. (vii) The head word of the preposition phrase that covers each verb, respectively.

To incorporate all the proposals in previous sections, first, we add features from TemProb which encodes prior knowledge of typical temporal orders of events into the feature set above; with these features, we also adopt the constraint-driven learning algorithm for TempRel classification; then CogCompTime assigns local prediction scores (i.e., soft-max scores) to each edge and solves an integer linear programming (ILP) problem via GUROBI [75] to achieve globally consistent temporal graphs.

Finally, MATRES proposed in Sec. 4.4 is selected as the evaluation benchmark.

The performance of TempRel extraction is P=61.6, R=70.9, F_1=65.9 when the gold event extraction is used. As a reference point, the best system in the TempEval3 workshop, ClearTK [98], had P=37.32, R=35.25, F_1=36.26 (using gold event extraction). Again, given the dataset difference, these numbers are not directly comparable, but it indicates that the proposals made in this chapter are indeed useful for the TempRel task.

4.5.2 Futher Improvement by Neural Methods

Since TempRel is a specific relation type, it is natural to borrow recent neural relation extraction approaches [135, 136, 137, 138]. There have indeed been such attempts, e.g., in clinical narratives [139, 140, 141] and in newswire [142, 143, 144]. However, their improvements over feature-based methods were moderate ([140] even showed negative results). We think it is important for us to understand: Is it because we are missing a "magic" neural architecture, because the training dataset is small, or because the quality of the dataset should be improved?

MATRES is relatively small (14K TempRels), but has a higher annotation quality from its improved task definition and annotation guideline. Now, we are in the right position to pursue neural methods for TempRel extraction. This subsection shows that a long short-term memory (LSTM) [145] system can readily outperform CogCompTime (feature-based) by a large margin; in contrast, the new system is called CogCompTime (neural-based). The fact that a standard LSTM system can significantly improve over a feature-based system on MATRES indicates that neural approaches have been mainly dwarfed by the quality of annotation, instead of specific neural architectures or the small size of data.

To gain a better understanding of the standard LSTM method, we extensively compare the usage of various word embedding techniques, including word2vec [146], GloVe [147], FastText [148], ELMo [149], and BERT [150], and show their impact on TempRel extraction. Moreover, we further improve the LSTM system by injecting knowledge from an updated version of TemProb. Altogether, these components

improve over CogCompTime (feature-based) by about 10% in F_1 and accuracy.

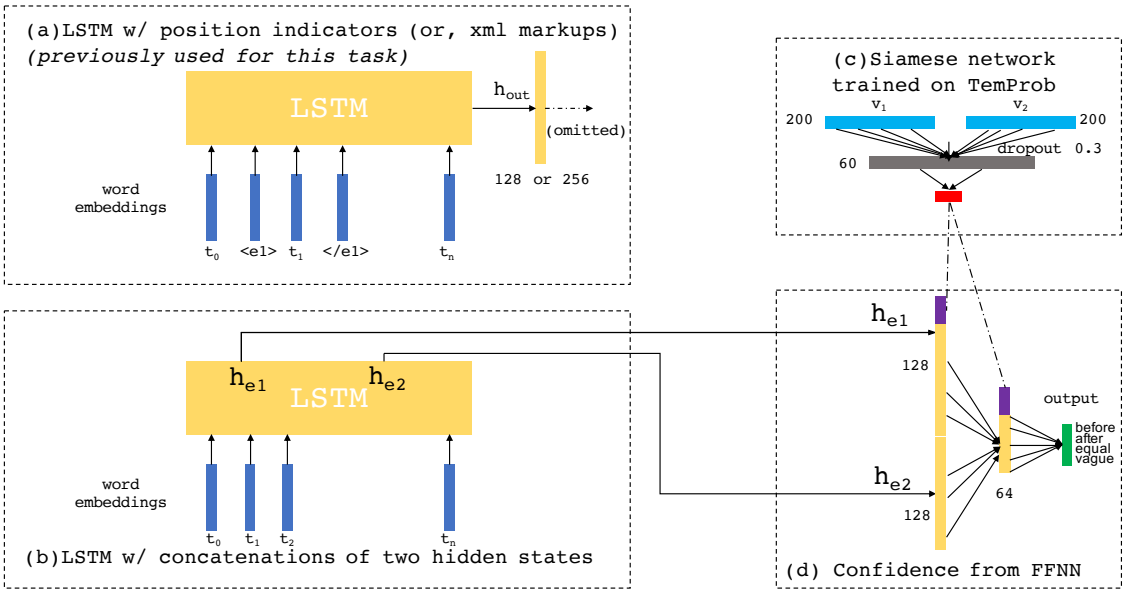


Figure 4.11: Overview of the neural network structures studied in this paper. Networks (a) and (b) are two ways to handle event positions in LSTMs (Sec. 4.5.2.1). (c) The Siamese network used to fit TemProb (Sec. 4.5.2.2). Once trained on TemProb, the Siamese network is fixed when training other parts of the system. (d) The FFNN that generates confidence scores for each label. Sizes of hidden layers are already noted. Embeddings of the same color share the same matrix.

One major disadvantage of feature-based systems is that errors which occurred in feature extraction propagate to subsequent modules. Here we study the usage of LSTM networks[18] on the TEMPREL extraction problem as an end-to-end approach that only takes a sequence of word embeddings as input (assuming that the position of events are known). Conceptually, we need to feed those word embeddings to LSTMs and obtain a vector representation for a particular pair of events, which is followed by a fully-connected, feed-forward neural network (FFNN) to generate confidence scores for each output label. Based on the confidence scores, global inference is performed

[18]We also tried convolutional neural networks but did not observe that CNNs improved performance significantly compared to the LSTMs. Comparison between LSTM and CNN is also not the focus of this paper.

via integer linear programming (ILP), which is a standard procedure used in many existing works to enforce the transitivity property of time [50, 92, 9]. An overview of the proposed network structure and corresponding parameters can be found in Fig. 4.11. Below we also explain the main components.

4.5.2.1 Handling Event Positions

Each TempRel is associated with two events, and for the same text, different pairs of events possess different relations, so it is critical to indicate the positions of those events when we train LSTMs for the task. The most straightforward way is to concatenate the hidden states from both time steps that correspond to the location of those events (Fig. 4.11b). Dligach et al. (2017) [139] handled this issue differently, by adding XML tags immediately before and after each event (Fig. 4.11a). For example, in the sentence, *After* ***eating*** *dinner, he* ***slept*** *comfortably*, where the two events are bold-faced, they will convert the sequence into *After <e1> eating </e1> dinner, he <e2> slept </e2> comfortably.* The XML markups, which were initially proposed under the name of *position indicators* for relation extraction [137], uniquely indicate the event positions to LSTM, such that the final output of LSTM can be used as a representation of those events and their context. We compare both methods in this paper, and as we show later, the straightforward concatenation method is already as good as XML tags for this task.

4.5.2.2 Common Sense Encoder (CSE)

In naturally occurring text that expresses TempRels, connective words such as *since*, *when*, or *until* are often not explicit; nevertheless, humans can still infer the TempRels using common sense with respect to the events. For example, even without context, we know that *die* is typically after *explode* and *schedule* typically before *attend*. Ning et al. (2018) [11] made initial attempt to acquire such knowledge by aggregating automatically extracted TempRels from a large corpus. The resulting knowledge base, TemProb, contains observed frequencies of tuples $(v1, v2, r)$ rep-

resenting the probability of verb 1 and verb 2 having relation r, and it was shown a useful resource for TempRel extraction.

However, TemProb is a simple counting model and fails (or is unreliable) for unseen (or rare) tuples. For example, we may see (ambush, die) less frequently than (attack, die) in a corpus, and the observed frequency of (ambush, die) being TempRelbefore or TempRelafter is thus less reliable. However, since "ambush" is semantically similar to "attack", the statistics of (attack, die) can actually serve as an auxiliary signal to (ambush, die). Motivated by this idea, we introduce common sense encoder (CSE): We fit an updated version of TemProb via a Siamese network [151] that generalizes to unseen tuples through the resulting embeddings for each verb (Fig. 4.11c). Note that the TemProb we use is reconstructed using the same method described in [11] with the base method changed to CogCompTime. Once trained, CSE will remain fixed when training the LSTM part (Fig. 4.11a or b) and the feedforward neural network part (Fig. 4.11d). We only use CSE for its output. In the beginning, we tried to directly use the output (i.e., a scalar) and the influence on performance was negligible. Therefore, here we discretize the CSE output, change it to categorical embeddings, concatenate them with the LSTM output, and then produce the confidence scores (Fig. 4.11d).

4.5.3 Experiments

4.5.3.1 Data

Some statistics of the datasets used in this section are shown in Table 4.21. The MATRES dataset contains 275 news articles from the TempEval3 workshop [3] with newly annotated events and TempRels. It has 3 sections: TimeBank (TB), AQUAINT (AQ), and Platinum (PT). We used the standard split (i.e., TB+AQ for training and PT for testing), and further set aside 20% of the training data as the development set to tune learning rates and epochs. We also show the performance

on another test set, TCR[19] [49], which contains both temporal and causal relations and we only need the temporal part. The label set for both datasets is *before*, *after*, *simultaneous*, and *vague*.

Table 4.21: TimeBank (TB), AQUAINT (AQ), and Platinum (PT) are from MATRES [12] and TCR from [49]. Note the numbers of events and TempRels do not match to those in Table 4.3 because Table 4.3 is for TempEval3 and here it is for MATRES.

	Purpose	#Doc	#Events	#TempRels
TB+AQ	Train	255	8K	13K
PT	Test	20	537	837
TCR	Test	25	1.3K	2.6K

4.5.3.2 Results

We compare CogCompTime (neural-based) with CogCompTime (feature-based), using three metrics for a more thorough comparison: classification accuracy (acc.), standard F_1, and temporal awareness F_{aware}, where the awareness score is for the graphs represented by a group of related TempRels (see Sec. 4.1.3). We also report the average of those three metrics in our experiments.

Table 4.22 compares the two different ways to handle event positions discussed in Sec. 4.5.2.1: position indicators (P.I.) and simple concatenation (Concat), both of which are followed by network (d) in Fig. 4.11 (i.e., without using Siamese yet). We extensively studied the usage of various pretrained word embeddings, including conventional embeddings (i.e., the medium versions of word2vec, GloVe, and FastText provided in the Magnitude package [152]) and contextualized embeddings (i.e., the original ELMo and large uncased BERT, respectively); except for the input embeddings, we kept all other parameters the same. We used cross-entropy loss and the StepLR optimizer in PyTorch that decays the learning rate by 0.5 every 10 epochs (performance not sensitive to it).

[19] `http://cogcomp.org/page/publication_view/835`

Comparing to the previously used P.I. [139], we find that, with only two exceptions (underlined in Table 4.22), the Concat system saw consistent gains under various embeddings and metrics. In addition, contextualized embeddings (ELMo and BERT) expectedly improved over the conventional ones significantly, although no statistically significant difference was observed between using ELMo or BERT.

Table 4.22: Performances on the MATRES test set (i.e., the PT section). CogCompTime [153] is the previous state-of-the-art feature-based system. Position indicator (P.I.) and concatenation (Concat) are two ways to handle event positions in LSTMs (Sec. 4.5.2.1). Concat+CSE (aka neural-based CogCompTime) achieves significant improvement over feature-based CogCompTime on MATRES.

System	*Emb.*	*Acc.*	F_1	F_{aware}	Avg.
P.I.	word2vec	63.2	67.6	<u>60.5</u>	63.8
	GloVe	64.5	69.0	<u>61.1</u>	64.9
	FastText	60.5	64.7	59.5	61.6
	ELMo	67.5	73.9	63.0	**68.1**
	BERT	68.8	73.6	61.7	68.0
Concat	word2vec	65.0	69.5	59.4	64.6
	GloVe	64.9	69.5	60.9	65.1
	FastText	64.0	68.6	60.1	64.2
	ELMo	67.7	74.0	63.3	68.3
	BERT	69.1	74.4	63.7	**69.1**
Concat+CSE	ELMo	71.7	76.7	66.0	**71.5**
	BERT	71.3	76.3	66.5	71.4
CogCompTime (feat.)	-	61.6	66.6	60.8	**63.0**

Given the above two observations, we further incorporated our common sense encoder (CSE) into "Concat" with ELMo and BERT in Table 4.22. We split TemProb into train (80%) and validation (20%). The proposed Siamese network (Fig. 4.11c) was trained by minimizing the cross-entropy loss using Adam [154] (learning rate 1e-4, 20 epochs, and batch size 500). We first see that CSE improved on top of Concat for both ELMo and BERT under all metrics, confirming the benefit of TemProb; second, as compared to CogCompTime (feature-based), the proposed Con-

cat+CSE achieved about 10% absolute gains in accuracy and F_1, 5% in awareness score F_{aware}, and 8% in the three-metric-average metric, with $p < 0.001$ per the McNemar's test. Roughly speaking, the 8% gain is contributed by LSTMs for 2%, contextualized embeddings for 4%, and CSE for 2%. Again, no statistically significant difference was observed between using ELMo and BERT. Table 4.23 furthermore applies CogCompTime (feature-based) and the proposed Concat+CSE system on a different test set called TCR. Both systems achieved better scores (suggesting that TCR is easier than MATRES), while the proposed system still outperformed CogCompTime (feature-based) by roughly 8% under the three-metric-average metric, consistent with our improvement on MATRES.

Table 4.23: Further evaluation of the proposed system, i.e., Concat (Sec. 4.5.2.1) plus CSE (Sec. 4.5.2.2), on the TCR dataset [49].

System	*Emb.*	*Acc.*	F_1	F_{aware}	Avg.
CogCompTime	-	68.1	70.7	61.6	**66.8**
Concat+CSE	ELMo	80.8	78.6	69.9	**76.4**
	BERT	78.4	77.0	69.0	74.9

4.5.3.3 Significance Test for Tables 4.22-4.23

In Table 4.22, we mainly compared the performance of position indicator (P.I.) and simple concatenation (Concat), using 5 different word embeddings and 3 metrics, so there were 15 performances for both P.I. and Concat. Under the paired t-test, Concat is significantly better than P.I. with $p < 0.01$.

Another observation we made in Table 4.22 was that contextualized embeddings, i.e., ELMo and BERT, were much better than conventional ones, i.e., word2vec, GloVe and FastText. For both P.I. and Concat, we found that the difference between contextualized embeddings and conventional embeddings was significant with $p < 0.001$ under the McNemar's test [117, 155]; however, between the two contextualized embeddings, ELMo and BERT, we did not see a significant difference, although it has been reported that in many *other* tasks, that BERT is better than ELMo.

In Table 4.23, we further improved Concat using the proposed common sense encoder (CSE). Under the McNemar's test, Concat+CSE was significantly better than Concat with $p < 0.001$, no matter whether ELMo or BERT was used. Again, no significant difference was observed between ELMo and BERT. Finally, since Concat+CSE improved over CogCompTime (feature-based) by a large margin either on MATRES or on TCR, it was not surprising to see that the proposed Concat+CSE is significantly better than CogCompTime (feature-based) with $p < 0.001$ as well.

CHAPTER 5

TEMPORAL COMMONSENSE UNDERSTANDING

Because people rarely say the obvious, natural language understanding requires the ability to reason with commonsense knowledge [156, 157], and the last few years have seen significant work in this direction (e.g., [158, 159, 160]). In terms of various temporal aspects of events, we have *temporal* common sense such as duration and frequency. However, this important problem has so far received limited attention. For instance, given two events *"going on a vacation"* and *"going for a walk"*, most humans would know that a vacation is typically longer and occurs less often than a walk, but it is still challenging for computers to understand and reason about temporal common sense. Therefore, in addition to TIMEX understanding and TEMPREL understanding, this thesis also systematically studies this *temporal common sense* problem.

Specifically, this thesis defines five classes of temporal common sense: *duration* (how long an event takes), *temporal ordering* (typical order of events), *typical time* (when an event happens), *frequency* (how often an event occurs), and *stationarity* (whether a state holds for a very long time or indefinitely). Existing works have investigated some of these aspects, either explicitly or implicitly (e.g., duration [161, 162] and ordering [122, 11]), but none of them have defined or studied all aspects of temporal common sense in a unified framework. Kozavera and Hovy (2011) [163] defined a few temporal aspects to be investigated, but failed to quantify performances on these phenomena.

Given the lack of evaluation standards and datasets for temporal common sense, this thesis develops a new dataset dedicated for it, MCTACO (short for **m**ultiple **c**hoice **t**empor**a**l **co**mmon sense). MCTACO is constructed via crowdsourcing with guidelines designed meticulously to guarantee its quality. When evaluated on MCTACO,

S1: *Growing up on a farm near St. Paul, L. Mark Bailey didn't dream of becoming a judge.*
Q1: *Is Mark still on the farm now?*
[x] no [ ] yes
Reasoning type: *stationarity*

S2: *The massive ice sheet, called a glacier, caused the features on the land you see today.*
Q2: *When did the glacier start to impact the land's features?*
[x] centuries ago [ ] hours ago
[ ] 10 years ago [x] tens of millions of years ago
Reasoning type: *event typical time*

S3: *Carl Laemmle, head of Universal Studios, gave Einstein a tour of his studio and introduced him to Chaplin.*
Q3: *How long did the tour last?*
[ ] 9 hours [ ] 15 days
[x] 45 minutes [ ] 5 seconds
Reasoning type: *event duration*

S4: *Mr. Barco has refused U.S. troops or advisers but has accepted U.S. military aid.*
Q4: *What happened after Mr. Barco accepted the military aid?*
[ ] the aid was denied [x] things started to progress
[x] he received the aid
Reasoning type: *event ordering*

S5: *The Minangkabau custom of freely electing their leaders provided the model for rulership elections in modern federal Malaysia.*
Q5: *How often are the elections held?*
[ ] every day [ ] every month
[x] every 4 years [ ] every 100 years
Reasoning type: *event frequency*

Figure 5.1: Five types of temporal common sense in MCTACO. Note that a question may have multiple correct answers.

a system receives a *sentence* providing context information, a *question* designed to require temporal common sense knowledge, and multiple *candidate answers* (see

Fig. 5.1). Note that in this setup, more than one candidate answer can be plausible, so the task is in fact a binary classification: determining whether a candidate answer is *plausible* according to human common sense. This is aligned with other efforts that have posed common sense as the choice of plausible alternatives [164]. The high quality of the resulting dataset (shown in Sec. 5.3) also makes us believe that the notion of plausibility here is robust.

Another finding is that, using MCTACO as a testbed, we study the temporal common sense understanding of the best existing NLP techniques, including *ESIM* [165], *BERT* [166] and their variants. Results in Sec. 5.3 show that, despite a significant improvement over random-guess baselines, the best existing techniques are still far behind human performance on temporal common sense understanding, indicating the need for further research in order to improve the currently limited capability to capture temporal semantics.

5.1 Related Work

Common sense has been a very popular topic in recent years and existing NLP works have mainly investigated the acquisition and evaluation of common sense in the physical world, including but not limited to, size, weight, and strength [167], roundness and deliciousness [168], and intensity [169]. In terms of "events" common sense, [170] investigated the intent and reaction of participants of an event, and [171] tried to select the most likely subsequent event. No earlier work has focused on *temporal* common sense, although it is critical for event understanding. For instance, [12] argues that resolving ambiguous and implicit mentions of event durations in text (a specific kind of temporal common sense) is necessary to construct the timeline of a story.

There have also been many works trying to understand time in natural language but not necessarily the commonsense understanding of time. Most recent works include the extraction and normalization of temporal expressions [4, 8], temporal relation extraction [9, 153], and timeline construction [144]. Among these, some

Table 5.1: Statistics of MCTACO.

Measure		Value
# of unique questions		1893
# of unique question-answer pairs		13,225
avg. sentence length		17.8
avg. question length		8.2
avg. answer length		3.3
Category	# questions	avg # of candidate
event frequency	433	8.5
event duration	440	9.4
event stationarity	279	3.1
event ordering	370	5.4
event typical time	371	6.8

works are implicitly on temporal common sense, such as event durations [162, 172], typical temporal ordering [122, 49, 11], and script learning (i.e., what happens next after certain events) [53, 55]. However, existing works have not studied all five types of temporal common sense in a unified framework as we do here, nor have they developed datasets for it.

Instead of working on each individual aspect of temporal common sense, this thesis formulates the problem as a machine reading comprehension task in the format of selecting plausible responses with respect to natural language queries. This relates our work to a large body of work on question-answering, an area that has seen significant progress in the past few years [173, 174, 175]. This area, however, has mainly focused on *general* natural language comprehension tasks, while we tailor it to test a *specific* reasoning capability, which is temporal common sense.

5.2 MCTACO: A Benchmark Dataset

MCTACO is comprised of 13K tuples, in the form of *(sentence, question, candidate answer)*; please see examples in Fig. 5.1 for the five phenomena studied here and

Table 5.1 for basic statistics of it. The sentences in those tuples are randomly selected from MultiRC [176] (from each of its 9 domains). For each sentence, crowdsourcing on Amazon Mechanical Turk was used to collect questions and candidate answers (both correct and wrong ones). To ensure the quality of the results, we limit the annotations to native speakers and use qualification tryouts.

5.2.1 Step 1: Question Generation

We first ask crowdsourcers to generate questions, given a sentence. To produce questions that need temporal common sense to answer, we require that a valid question: (a) should ask about one of the five temporal phenomena we defined earlier, and (b) should not be solved simply by a word or phrase from the original sentence. We also require crowdsourcers to provide a correct answer for each of their questions, which on one hand gives us a positive candidate answer, and on the other hand ensures that the questions are answerable at least by themselves.

5.2.2 Step 2: Question Verification

We further ask another two crowdsourcers to check the questions generated in Step 1, i.e., (a) whether the two requirements are satisfied and (b) whether the question is grammatically and logically correct. We retain only the questions where the two annotators unanimously agree with each other and the decision generated in Step 1. For valid questions, we continue to ask crowdsourcers to give one correct answer and one incorrect answer, which we treat as a seed set to automatically generate new candidate answers in the next step.

5.2.3 Step 3: Candidate Answer Expansion

Until this stage, we have collected a small set of candidate answers (3 positive and 2 negative) for each question.[1] We automatically expand this set in three ways. First, we use a set of rules to extract numbers and quantities ("2", "once") and temporal terms (e.g. "a.m.", "1990", "afternoon", "day"), and then randomly perturb them based on a list of temporal units ("second"), adjectives ("early"), points ("a.m.") and adverbs ("always"). Examples are "2 a.m." → "3 p.m.", "1 day" → "10 days", "once a week"→ "twice a month" (more details in the appendix).

Second, we mask each individual token in a candidate answer (one at a time) and use *BERT* [166] to predict replacements for each missing term; we rank those predictions by the confidence level of *BERT* and keep the top three.

Third, for those candidates that represent events, the previously-mentioned token-level perturbations rarely lead to interesting and diverse set of candidate answers. Furthermore, it may lead to invalid phrases (e.g., "he left the house" → "he walked the house"). Therefore, to perturb such candidates, we create a pool of $60k$ event phrases using PropBank [37], and perturb the candidate answers to be the most similar ones extracted by an information retrieval (*IR*) system.[2] This not only guarantees that all candidates are properly phrased, it also leads to more diverse perturbations.

We apply the above three techniques on non-"event" candidates sequentially, in the order they were explained, to expand the candidate answer set to 20 candidates per question. A perturbation technique is used, as long as the pool of candidates is still less than 20. Note there are both correct and incorrect answers in those candidates.

5.2.4 Step 4: Answer Labeling

In this step, each (*sentence, question, answer*) tuple produced earlier is labeled by 4 crowdsourcers, with three options: "likely", "unlikely", or "invalid" (sanity check for

[1] One positive answer from Step 1; one positive and one negative answer from each of the two annotators in Step 2.

[2] `www.elastic.co`

valid tuples).[3] Different annotators may have different interpretations, yet we ensure label validity through high agreement. A tuple is kept only if all 4 annotators agree on "likely" or "unlikely". The final statistics of MCTACO is in Table 5.1.

5.3 Experiments

We assess the quality of our dataset through human annotation, and evaluate a couple of baseline systems. We create a uniform split of 30%/70% of the data to dev/test. The rationale behind this split is that a successful system has to bring in a huge amount of world knowledge and derive commonsense understandings *prior* to the current task evaluation. We therefore believe that it is not reasonable to expect a system to be *trained* solely on this data, and we think of the development data as only providing a *definition* of the task. Indeed, the gains from our development data are marginal after a certain number of training instances. This intuition is studied and verified in Sec. 5.4.2.

5.3.1 Evaluation Metrics

Two question-level metrics are adopted in this work: exact match (EM) and *F1*. For a given candidate answer a that belongs to a question q, let $f(a;q) \in \{0,1\}$ denote the correctness of the prediction made by a fixed system (1 for correct; 0 otherwise). Additionally, let D denote the collection of questions in our evaluation set.

$$EM \triangleq \frac{\sum_{q \in D} \prod_{a \in q} f(a;q)}{|\{q \in D\}|}.$$

The recall for each question q is:

$$R(q) = \frac{\sum_{a \in q} [f(a;q) = 1] \wedge [a \text{ is "likely" }]}{|\{a \text{ is "likely" } \wedge a \in q\}|}.$$

[3] We use the name "(un)likely" because commonsense decisions can be naturally ambiguous and subjective.

Similarly, $P(q)$ and $F1(q)$ are defined. The aggregate $F1$ (across the dataset D) is the macro average of question-level $F1$'s:

$$F1 \triangleq \frac{\sum_{q \in D} F1(q)}{|\{q \in D\}|}.$$

EM measures how many questions for which a system is able to correctly label all candidate answers, while *F1* is more relaxed and measures the average overlap between one's predictions and the ground truth.

5.3.2 Human Performance

An expert annotator also worked on MCTACO to gain a better understanding of the human performance on it. The expert answered 100 questions (about 700 (*sentence, question, answer*) tuples) randomly sampled from the test set, and could only see a single answer at a time, with its corresponding question and sentence.

5.3.3 Systems

We use two state-of-the-art systems in machine reading comprehension for this task: *ESIM* [165] and *BERT* [166]. *ESIM* is an effective neural model on natural language inference. We initialize the word embeddings in *ESIM* via either *GloVe* [147] or *ELMo* [149] to demonstrate the effect of pre-training. *BERT* is a state-of-the-art contextualized representation used for a broad range of tasks . We also add unit normalization to *BERT*, which extracts and converts temporal expressions in candidate answers to their most proper units. For example, "30 months" will be converted to "2.5 years". To the best of our knowledge, there are no other available systems for the "stationarity", "typical time", and "frequency" phenomena studied here. As for "duration" and "temporal order", there are existing systems (e.g., [172, 11]), but they cannot be directly applied to the setting in MCTACO where the inputs are natural languages.

Table 5.2: Summary of the performances for different baselines. All numbers are percentages.

System	*F1*	*EM*
Random	36.2	8.1
Always Positive	49.8	12.1
Always Negative	17.4	17.4
ESIM + GloVe	50.3	20.9
ESIM + ELMo	54.9	26.4
BERT	66.1	39.6
BERT + unit normalization	**69.9**	**42.7**
Human	87.1	75.8

5.3.4 Experimental Setting

In both *ESIM* baselines, we model the process as a sentence-pair classification task, following the *SNLI* setting in AllenNLP.[4] In both versions of *BERT*, we use the same sequence pair classification model and the same parameters as in *BERT*'s *GLUE* experiments.[5] A system receives two elements at a time: (a) the concatenation of the sentence and question, and (b) the answer. The system makes a binary prediction on each instance, "likely" or "unlikely".

5.3.5 Results

Table 5.2 compares native baselines, *ESIM*, *BERT* and their variants on the entire test set of MCTACO; it also shows human performance on the subset of 100 questions.[6] The system performances reported are based on default random seeds, and we observe a maximum standard error[7] of 0.8 from 3 runs on different seeds

[4] `https://github.com/allenai/allennlp`

[5] `https://github.com/huggingface/pytorch-pretrained-BERT`

[6] *BERT + unit normalization* scored $F1 = 72, EM = 45$ on this subset, which is only slightly different from the corresponding number on the entire test set.

[7] `https://en.wikipedia.org/wiki/Standard_error`

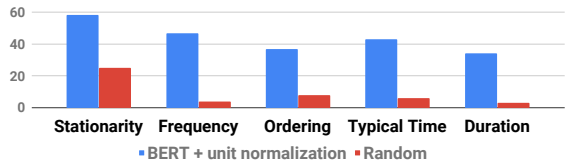


Figure 5.2: *EM* scores of *BERT + unit normalization* per temporal reasoning category comparing to the random-guess baseline.

across all entries. We can confirm the good quality of this dataset based on the high performance of human annotators. *ELMo* and *BERT* improve naive baselines by a large margin, indicating that a notable amount of commonsense knowledge has been acquired via pre-training. However, even *BERT* still falls far behind human performance, indicating the need of further research. For example, RoBERTa [177], a more recent language model that was released recently, achieves $F1 = 72.3, EM = 43.6$.

We know that *BERT*, as a language model, is good at associating surface forms (e.g. associating "sunrise" with "morning" since they often co-occur), but may be brittle with respect to variability of temporal mentions.

Consider the following example (the correct answers are indicated with ✓and *BERT* selections are underlined.) This is an example of *BERT* correctly associating a given event with "minute" or "hour"; however, it fails to distinguish between "1 hour" (a "likely" candidate) and "9 hours" (an "unlikely" candidate).

P: *Ratners's chairman, Gerald Ratner, said the deal remains of "substantial benefit to Ratners."*
Q: *How long did the chairman speak?*
✓(a) 30 minutes ✓(b) 1 hour
✗(c) 9 hours ✗(d) twenty seconds

This shows that *BERT* does not infer a range of true answers; it instead associates discrete terms and decides individual options, which may not be the best way to handle temporal units that involve numerical values.

BERT+unit normalization is used to address this issue, but results show that it

is still poor compared to human. This indicates that the information acquired by *BERT* is still far from solving temporal common sense.

Since exact match (EM) is a stricter metric, it is consistently lower than $F1$ in Table 5.2. For an ideal system, the gap between EM and $F1$ should be small (humans only drop 11.3%.) However, all other systems drop by almost 30% from $F1$ to EM, possibly another piece of evidence that they only associate surface forms instead of using one representation for temporal common sense to classify all candidates.

A curious reader might ask why the human performance on this task as shown in Table 5.2 is not 100%. This is expected because common sense is what *most* people agree on, so any *single* human could disagree with the gold labels in MCTACO. Therefore, we think the human performance in Table 5.2 from a single evaluator actually indicates the good quality of MCTACO.

The performance of *BERT+unit normalization* is not uniform across different categories (Fig. 5.2), which could be due to the different nature or quality of data for those temporal phenomena. For example, as shown in Table 5.1, "stationarity" questions have much fewer candidates and a higher random baseline.

5.4 Discussion

5.4.1 Perturbing Candidate Answers

Here we provide a few missing details from *Step 3* of our annotations (Sec. 5.2.3). In particular, we create collections of common temporal expressions (see Table 5.3) to detect whether the given candidate answer contains a temporal expression or not. If a match is found within this list, we use the mappings to create perturbations of the temporal expression.

Table 5.3: Collections of temporal expressions used in creating perturbation of the candidate answers. Each mention is grouped with its variations (e.g., "first" and "last" are in the same set).

Adjectives	Frequency	Period	Typical time	Units
early:late	always:sometimes:never	night:day	now:later	second:hour:week:year
late:early	occasionally:always:never	day:night	today:yesterday	seconds:hours:weeks:years
morning:late night	often:rarely		tomorrow:yesterday	minute:day:month:century
night:early morning	usually:rarely		tonight:last night	minutes:days:months:centuries
evening:morning	rarely:always		yesterday:tomorrow	hour:second:week:year
everlasting:periodic	constantly:sometimes		am:pm	hours:seconds:weeks:years
initial:last	never:sometimes:always		pm:am	day:minute:month:century
first:last	regularly:occasionally:never		a.m.:p.m.	days:minutes:months:centuries
last:first			p.m.:a.m.	week:second:hour:year
overdue:on time			afternoon:morning	weeks:seconds:hours:years
belated:punctual			morning:evening	month:minute:day:century
long-term:short-term			night:morning	months:minutes:days:centuries
delayed:early			after:before	year:second:hour:week
punctual:belated			before:after	years:seconds:hours:weeks
				century:minute:day:month
				centuries:minutes:days:months

5.4.2 Performance as a Function of Training Size

An intuition that we stated is that the task at hand requires a successful model to bring in external world knowledge beyond what is observed in the dataset; for a task like this, it is unlikely to compile a dataset which covers all the possible events and their attributes. In other words, the "traditional" supervised learning alone (with no pre-training or external training) is unlikely to succeed. A corollary to this observation is that tuning a pre-training system (such as BERT [166]) likely requires very little supervision.

We plot the performance change, as a function of number of instances observed in the training time (Fig. 5.3). Each point in the figure shares the same parameters and averages of 5 distinct trials over different random sub-samples of the dataset. As can be observed, the performance plateaus after about $2.5k$ question-answer pairs (about 20% of the whole datasets). This verifies the intuition that systems can rely on a relatively small amount of supervision to tune to task, if it models the world knowledge through pre-training. Moreover, it shows that trying to make improvement through getting more labeled data is costly and impractical.

In summary, this chapter has focused on temporal common sense. We define five

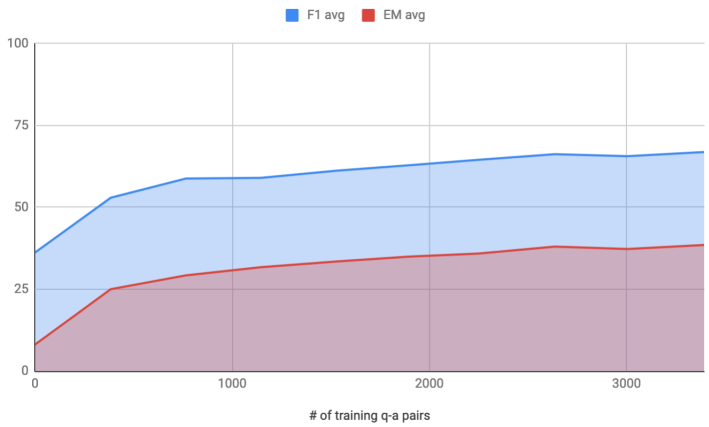


Figure 5.3: Performance of supervised algorithm (BERT; Section 4) as function of various sizes of observed training data. When no training data is provided to the systems (left-most side of the figure), the performance measures amount to random guessing.

categories of questions that require temporal common sense and develop a novel crowdsourcing scheme to generate MCTACO, a high-quality dataset for this task. We use MCTACO to probe the capability of systems on temporal common sense understanding. We find that systems equipped with state-of-the-art language models such as *ELMo* and *BERT* are still far behind humans, thus motivating future research in this area. Our analysis sheds light on the capabilities as well as limitations of current models.

CHAPTER 6

INVESTIGATION OF INCIDENTAL SUPERVISION

For many structured learning tasks (including the temporal tasks extensively studied in this thesis), the data annotation process is complex and costly. Existing annotation schemes usually aim at acquiring *completely* annotated structures, under the common perception that *partial* structures are of low quality and could hurt the learning process. This chapter questions this common perception, motivated by the fact that structures consist of *interdependent* sets of variables. Thus, given a fixed budget, partly annotating each structure may provide the same level of supervision, while allowing for more structures to be annotated. We provide an information theoretic formulation for this perspective and use it, in the context of both the TempRel extraction task and another two structured learning tasks, to show that learning from partial structures can *sometimes* outperform learning from complete ones. The important findings here may provide important insights into structured data annotation schemes and could support progress in learning protocols for structured tasks.

6.1 Motivation

Many machine learning tasks require structured outputs, and the goal is to assign values to a set of variables coherently. Specifically, the variables in a structure need to satisfy some global properties required by the task. An important implication is that once some variables are determined, the values taken by other variables are constrained. For instance, in the temporal relation extraction problem in Fig. 6.1a, if *met* happened before *leaving* and *leaving* happened on *Thursday*, then we know

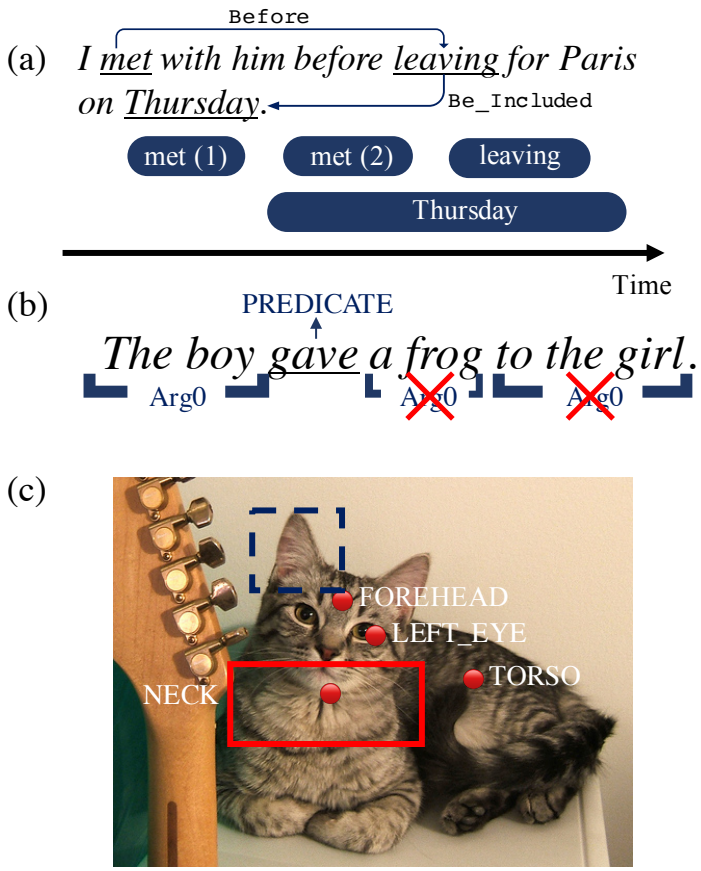


Figure 6.1: Due to the inherent structural constraints of each task, individual instances therein put restrictions on others. (a) The temporal relation between *met* and *Thursday* has to be *Before* ("met (1)") or *Be_Included* ("met (2)"). (b) The argument roles of *a frog* and *to the girl* cannot be *Arg0* anymore. (c) Given the position of the cat's FOREHEAD and LEFT_EYE, a rough estimate of its NECK can be the red solid box rather than the blue dashed box.

that *met* must either be before *Thursday* ("met (1)") or has to happen on *Thursday*, too ("met (2)") [49]. Similarly, in the semantic frame of the predicate *gave* [37] in Fig. 6.1b, if *the boy* is *Arg0* (short for argument 0), then it rules out the possibility of *a frog* or *to the girl* taking the same role. Figure 6.1c further shows an example of part-labeling of images [178]; given the position of FOREHEAD and LEFT_EYE of the cat in the picture, we roughly know that its NECK should be somewhere in the

red solid box, while the blue dashed box is likely to be wrong.

Data annotation for these structured tasks is complex and costly, thus requiring one to make the most of a given budget. This issue has been investigated for decades from the perspective of active learning for classification tasks [179, 180, 181] and for structured tasks [182, 183, 184, 185]. While active learning aims at selecting the next structure to label, we try to investigate, from a different perspective, whether we should annotate each structure completely or partially. Conventional annotation schemes typically require *complete* structures, under the common perception that partial annotation could adversely affect the performance of the learning algorithm. But note that partial annotations will allow for more structures to be annotated (see Fig. 6.2). Therefore, a fair comparison should be done while maintaining a fixed annotation budget, which was not done before. Moreover, even if partial annotation leads to comparable learning performance to conventional complete schemes, it provides more flexibility in data annotation.

Another potential benefit of partial annotation is that it imposes constraints on the remaining parts of a structure. As illustrated by Fig. 6.1, with partial annotations, we already have some knowledge about the unannotated parts. Therefore, further annotations of these variables may use the available budget less efficiently; this effect was first discussed in [10]. Motivated by the observations in Figs. 6.1-6.2, we think it is important to study partialness systematically, before we hastily assume that *completeness* should always be favored in data collection.

To study whether the above benefits of partialness can offset its weakness for learning, this thesis proposes the *early stopping partial annotation* (ESPA) scheme, which randomly picks up instances to label in the beginning, and stops before a structure is completed. We do not claim that ESPA should *always* be preferred; instead, it serves as an *alternative* to conventional, complete annotation schemes that we should keep in mind, because, as we show later, it can be comparable to (and sometimes even better than) complete annotation schemes. ESPA is straightforward to implement even in crowdsourcing; instances to annotate can be selected *offline* and distributed to crowdsourcers; this can be contrasted with the difficulties of implementing active learning protocols in these settings [186, 187]. We think that ESPA is a good

representative for a systematic study of partialness.

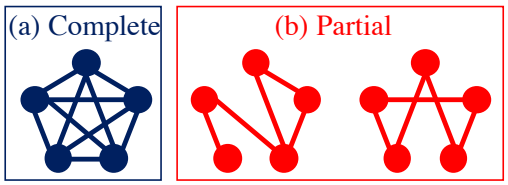


Figure 6.2: If we need training data for a graph labeling task (assuming the gold values for the nodes are given) and our annotation budget allows us to annotate, for instance, 10 edges in total, we could (a) completely annotate one graph (and then we run out of budget), or (b) partially annotate two graphs.

This thesis also develops an information theoretic formulation to explain the benefit of ESPA (Sec. 6.2), which we further demonstrate via three structured learning tasks in Sec. 6.4: temporal relation (TempRel) extraction [3], semantic role classification (SRC),[1] and shallow parsing [188]. These tasks are chosen because they each represent a wide spectrum of structures that we will detail later. As a byproduct, we extend CoDL [112] to cope with partially annotated structures (Sec. 6.3); we call the algorithm *Structured Self-learning with Partial ANnotations* (SSPAN) to distinguish it from CoDL.[2]

We believe in the importance of this chapter. First, partialness is inevitable in practice, either by mistake or by choice, so our theoretical analysis can provide unique insight into understanding partialness. Second, it opens up opportunities for new annotation schemes. Instead of considering partial annotations as a compromise, we can in fact annotate partial data *intentionally*, allowing us to design favorable guidelines and collect more important annotations at a cheaper price. Many recent datasets that were collected via crowdsourcing are already partial, and this chapter provides some theoretical foundations for them. Furthermore, the setting described here addresses natural scenarios where only partial, indirect supervision is available,

[1] A subtask of semantic role labeling (SRL) [34] that only classifies the role of an argument.

[2] There have been many works on learning from partial annotations, which we review in Sec. 6.3. SSPAN is only an experimental choice in demonstrating ESPA. Whether SSPAN is better than other algorithms is out of the scope here, and a better algorithm for ESPA will only strengthen the claims in this chapter.

as in *Incidental Supervision* [189], and this chapter begins to provide theoretical understanding for this paradigm, too. Further discussions can be found in Sec. 6.5.

It is important to clarify that we assume uniform cost over individual annotations (that is, all edges in Fig. 6.2 cost equally), often the default setting in crowdsourcing. We realize that the annotation difficulty can vary a lot in practice, sometimes incurring different costs. To address this issue, we randomly select instances to label so that on average, the cost is uniform. We agree that, even with this randomness, there could still be situations where the assumption does not hold, but we leave it for future studies, possibly in the context of active learning schemes.

6.2 Early Stopping Partial Annotation

In this section, we study whether the effect demonstrated by the examples in Fig. 6.1 exists in general. First, recall the definition of *structure* (Definition 1 in Sec. 4.2); it is necessary to model a structure as a set of *random* variables because when it is not completely annotated, there is still uncertainty in the annotation assignment. Intuitively, annotations are essentially reducing this uncertainty by labeling its variables. To study partial annotations, here we formally define *annotation* as well.

Definition 2 *A k-step annotation ($0 \leq k \leq d$) is a vector of RVs $\mathbf{A}_k = [A_{k,1}, \ldots, A_{k,d}] \in (\mathcal{L} \cup \sqcap)^d$ where $\sqcap$ is a special character for null, such that*

$$\sum_{i=1}^{d} \mathbb{1}(A_{k,i} \neq \sqcap) = k, \tag{6.1}$$

$$P(\mathbf{Y}|\mathbf{A}_k = \mathbf{a}_k) = P(\mathbf{Y}|Y_j = a_{k,j}, j \in \mathcal{J}), \tag{6.2}$$

where $\mathcal{J}$ is the set of indices that $a_{k,j} \neq \sqcap$.

Equation (6.1) means that, in total, k variables are already annotated at step k. Obviously, $\mathbf{A}_0$ means that no variables are labeled, and $\mathbf{A}_d$ means that all variables in $\mathbf{Y}$ are determined. $\mathbf{A}_k$ is what we call a k-step ESPA, so hereafter we use k/d to

represent annotation completeness. Equation (6.2) assumes no annotation mistakes, so if the i-th variable is labeled, then Y_i must be the same as $A_{k,i}$.

To measure the theoretical benefit of $\mathbf{A}_k$, we propose the following quantity:

$$I_k = \log |C(\mathcal{L}^d)| - E\left[\log f(\mathbf{a}_k)\right] \tag{6.3}$$

for $k = 0, \ldots, d$, where $f(\mathbf{a}_k) = |\{\mathbf{y} \in C(\mathcal{L}^d) : \mathrm{P}\left(\mathbf{y}|\mathbf{a}_k\right) > 0\}|$ is the total number of structures in $C(\mathcal{L}^d)$ that are still valid given $\mathbf{A}_k = \mathbf{a}_k$. Since we assume that the labeled variables in $\mathbf{A}_k$ are selected uniformly randomly, $E\left[\cdot\right]$ is simply the average of $\log f(\mathbf{a}_k)$. When $k = 0$, $f(\mathbf{a}_k) \equiv C(\mathcal{L}^d)$ and $I_0 \equiv 0$; as k increases, I_k increases since the structure has more and more variables labeled; finally, when $k = d$, the structure is fully determined and $I_d \equiv \log |C(\mathcal{L}^d)|$. The first-order finite difference, $I_k - I_{k-1}$, is the benefit brought by annotating an additional variable at step k; if I_k is concave (i.e., a decaying $I_k - I_{k-1}$), the benefit from a new annotation attenuates, suggesting the potential benefit of the ESPA strategy.

In an extreme case where the structure is so strong that it requires all individual variables to share the same label, then labeling any variable is sufficient for determining the entire structure. Intuitively, we do not need to annotate more than one variable. Our I_k quantity can support this intuition: The structural constraint, $C(\mathcal{L}^d)$, contains only $|\mathcal{L}|$ elements: $\{[\ell_i, \ell_i, \ldots, \ell_i]\}_{i=1}^{|\mathcal{L}|}$, so $I_0 = 0$, and $I_1 = \cdots = I_d = \log |\mathcal{L}|$. Since I_k does not increase at all when $k >= 1$, we should adopt first-step annotation $\mathbf{A}_1$. Another extreme case is that of a trivial structure that has no constraints (i.e., $C(\mathcal{Y}^d) = \mathcal{Y}^d$). The annotations of all variables are independent and we gain no advantage from skipping any variables. This intuition can be supported by our I_k analysis as well: Since $I_k = k \log |\mathcal{L}|$, $\forall k = 0, 1, \ldots, d$, I_k is linear and all steps contribute equally to improving I_k by $\log |\mathcal{L}|$; therefore ESPA is not necessary.

Real-world structures are often not as trivial as the two extreme cases above, but I_k can still serve as a guideline to help determine whether it is beneficial to use ESPA. We next discuss three diverse types of structures and how to obtain I_k for them.

Example 23: *The ranking problem is an important machine learning task and often depends on pairwise comparisons, for which the label set is* $\mathcal{L} = \{<, >\}$. *For*

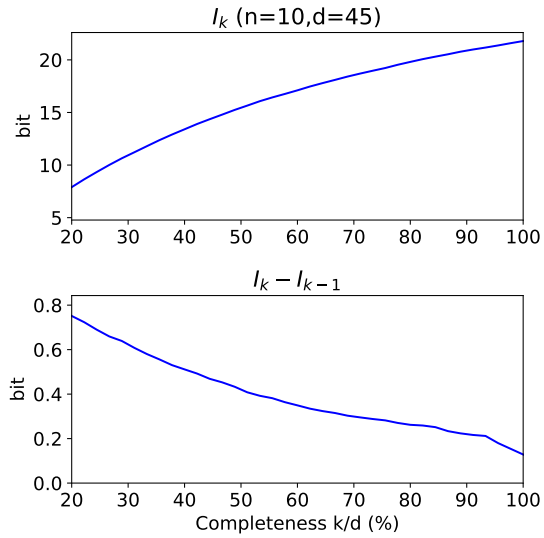


Figure 6.3: The mutual information between the chain structure and its k-step ESPA, I_k, is concave, suggesting possible benefit of using ESPA. In the simulation, there are $n = 10$ items in the chain and thus $d = 45$ pairs, k of which are labeled. The values of I_k's, as defined by Eq. (6.3), were obtained through averaging 1000 experiments. We use base-2 logarithm and the unit on y-axis is thus "bit".

a ranking problem with n items, there are $d = n(n-1)/2$ pairwise comparisons in total. Its structure is a chain following the transitivity constraints, i.e., if $A < B$ and $B < C$, then $A < C$.The ranking problem is an important machine learning task and often depends on pairwise comparisons, for which the label set is $\mathcal{L} = \{<, >\}$. For a ranking problem with n items, there are $d = n(n-1)/2$ pairwise comparisons in total. Its structure is a chain following the transitivity constraints, i.e., if $A < B$ and $B < C$, then $A < C$.

A k-step ESPA $\mathbf{A}_k$ for a chain means that only k (out of d) pairs are compared and labeled, resulting in a directed acyclic graph (DAG). In this case, $f(\mathbf{a}_k)$ is actually counting the number of linear extensions of the DAG, which is known to be #P-complete [190], so we do not have a closed-form solution to I_k. In practice, however, we can use the Kahn's algorithm and backtracking to simulate with a relatively small n, as shown by Fig. 6.3, where $n = 10$ and I_k was obtained through averaging

1000 random simulations. I_k is concave, as reflected by the downward shape of $I_k - I_{k-1}$. Therefore, new annotations are less and less efficient for the chain structure, suggesting the usage of ESPA.

Example 24: *The general assignment problem requires assigning d agents to d' tasks such that the agent nodes and the task nodes form a bipartite graph (without loss of generality, assume $d \leq d'$). That is, an agent can handle exactly one task, and each task can only be handled by at most one agent. Then from the agents' point of view, the label set for each of them is $\mathcal{L} = \{1, 2, \ldots, d'\}$, denoting the task assigned to the agent. Note that the classic M-ary classification problem is a special case with $d = 1$ and $d' = M$.*

A k-step ESPA $\mathbf{A}_k$ for this problem means that k agents are already assigned with tasks, and $f(\mathbf{a}_k)$ is to count the valid assignments of the remaining tasks to the remaining $d-k$ agents, to which we have closed-form solutions: $f(\mathbf{a}_k) = \frac{(d'-k)!}{(d'-d)!}$, $\forall \mathbf{a}_k$. According to Eq. (6.3), $I_k = \log \frac{d'!}{(d'-k)!}$ regardless of d or the distribution of $\mathbf{A}_k$, and is concave (Fig. 6.4 shows an example of it when $d = 4, d' = 10$).

Example 25: *Sequence tagging is an important NLP problem, where the tags of tokens are interdependent. Take chunking as an example. A basic scheme is for each token to choose from three labels, B(egin), I(nside), and O(utside), to represent text chunks in a sentence. That is, $\mathcal{L} = \{B, I, O\}$. Obviously, O cannot be immediately followed by I.*

Let d be the number of tokens in a sentence. A k-step ESPA $\mathbf{A}_k$ for chunking means that k tokens are already labeled by B/I/O, and $f(\mathbf{a}_k)$ counts the valid BIO sequences that do not violate those existing annotations. Again, as far as we know, there is no closed-form solution to $f(\mathbf{a}_k)$ and I_k, but in practice, we can use dynamic programming to obtain $f(\mathbf{a}_k)$ and then I_k using Eq. (6.3). We set $d = 10$ and show $I_k - I_{k-1}$ for this task in Fig. 6.4, where we observe the same effect we see in previous examples: The benefit provided by labeling a new token in the structure attenuates.

Interestingly, based on Fig. 6.4, we find that the slope of $I_k - I_{k-1}$ may be a good measure of the "tightness" or "strength" of a structure. When there is no structure at all, the curve is flat (black). The BIO structure is intuitively simple, and it indeed has the flattest slope among the three structured tasks (purple). When the structure

is a chain, the level of uncertainty goes down rapidly with every single annotation (think of standard sorting algorithms); the constraint is intuitively strong and in Fig. 6.4, it indeed has a steep slope (blue).

Finally, we want to emphasize that the definition of I_k in Eq. (6.3) is in fact backed by information theory. When we do not have prior information about $\mathbf{Y}$, we can assume that $\mathbf{Y}$ follows a uniform distribution over $C(\mathcal{L}^d)$. Then, I_k is essentially the mutual information between structure $\mathbf{Y}$ and annotation $\mathbf{A}_k$, $I(\mathbf{Y}; \mathbf{A}_k)$:

$$\begin{aligned} I(\mathbf{Y}; \mathbf{A}_k) &= H(\mathbf{Y}) - H(\mathbf{Y}|\mathbf{A}_k) \\ &= \log |C(\mathcal{L}^d)| - E\left[H(\mathbf{Y}|\mathbf{A}_k = \mathbf{a}_k)\right] \\ &= \log |C(\mathcal{L}^d)| - E\left[\log f(\mathbf{a}_k)\right], \end{aligned}$$

where $H(\cdot)$ is the entropy function. This is an important discovery, since it points out a new way to view a structure and its annotations. It may be useful for studying active learning methods for structured tasks, and other annotation phenomena such as noisy annotations. The usage of mutual information also aligns well with the information bottleneck framework [191, 192, 193], although a more recent paper challenges the interpretation of information bottleneck [194].

6.3 Learning from Partial Structures

So far, we have been advocating the ESPA strategy to maximize the information we can get from a fixed budget. Since early stopping leads to partial annotations, one missing component before we can benefit from it is an approach to learning from partial structures. In this study, we assume the existence of a relatively small but complete dataset that can provide a good initialization for learning from a partial dataset, which is very similar to semi-supervised learning (SSL). SSL, in its most standard form, studies the combined usage of a labeled set $\mathcal{T} = \{(x_i, y_i)\}_i$ and an unlabeled set $\mathcal{U} = \{x_j\}_j$, where the x's are instances and y's are the corresponding labels. SSL gains information about $p(x)$ through $\mathcal{U}$, which may improve the estima-

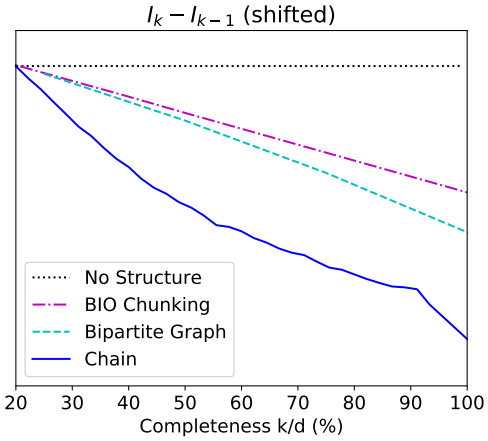


Figure 6.4: The $I_k - I_{k-1}$ curves from several different structures. The curves are shifted to almost the same starting point for better visualization, so the Y-Axis grid is not shown. The curve for "Chain" was obtained via simulations, and the other curves all have closed-form formulations.

tion of $p(y|x)$. Specific algorithms range from self-training [195, 196], to co-training [197], generative models [198], transductive SVM [199], etc., among which one of the most basic algorithms is Expectation-Maximization (EM) [200]. By treating them as hidden variables, EM "marginalizes" out the missing labels of $\mathcal{U}$ via expectation (i.e., soft EM) or maximization (i.e., hard EM). For structured ML tasks, soft and hard EMs turn into posterior regularization (PR) [201] and constraint-driven learning (CoDL) [112], respectively.

Unlike unlabeled data, the partially annotated structures caused by early stopping urge us to gain information not only about $p(x)$, but also from their labeled parts. There are many works along this line [202, 203, 204, 205], but in this thesis, we decide to extend CoDL to cope with partial annotations for two reasons. First, CoDL, which itself can be viewed as an extension of self-training to *structured* learning, is a wrapper algorithm having wide applications. Second, as its name suggests, CoDL learns from $\mathcal{U}$ by guidance of constraints, so partial annotations in $\mathcal{U}$ are technically easy to be added as extra equality constraints.

Algorithm 3 describes our *Structured Self-learning with Partial ANnotations* (SS-

PAN) algorithm that learns a model $\mathcal{H}$. The same as CoDL, SSPAN is a wrapper algorithm requiring two components: LEARN and INFERENCE. LEARN attempts to estimate the *local* decision function for each individual instance regardless of the *global* constraints, while INFERENCE takes those local decisions and performs a *global* inference. Lines 3-9 are the procedure of self-training, which iteratively completes the missing annotations in $\mathcal{P}$ and learns from both $\mathcal{T}$ and the completed version of $\mathcal{P}$ (i.e., $\tilde{\mathcal{P}}$).[3] Line 6 requires that the inference follows the structural constraints inherently in the task, turning the algorithm into CoDL; Line 7 enforces those partial annotations in $\mathbf{a}_i$, further turning it into SSPAN. In practice, INFERENCE can be realized by the Viterbi or beam search algorithm in sequence tagging, or more generally, by integer linear programming (ILP) [80]; either way, the partial constraints of Line 7 can be easily incorporated.

Algorithm 3: Structured Self-learning with Partial Annotations (SSPAN)

Input: $\mathcal{T} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$, $\mathcal{P} = \{(\mathbf{x}_i, \mathbf{a}_i)\}_{i=N+1}^{N+M}$

1 Initialize $\mathcal{H} = \text{LEARN}(\mathcal{T})$
2 **while** *convergence criteria not satisfied* **do**
3 $\tilde{\mathcal{P}} = \emptyset$
4 **foreach** $(\mathbf{x}_i, \mathbf{a}_i) \in \mathcal{P}$ **do**
5 $\hat{\mathbf{y}}_i = \text{INFERENCE}(\mathbf{x}_i; \mathcal{H})$, such that
6 $\diamond$ $\hat{\mathbf{y}}_i \in C(\mathcal{Y}^d)$
7 $\diamond$ $\hat{y}_{i,j} = a_{i,j}, \forall a_{i,j} \neq \sqcap$
8 $\tilde{\mathcal{P}} = \tilde{\mathcal{P}} \cup \{(\mathbf{x}_i, \hat{\mathbf{y}}_i)\}$
9 $\mathcal{H} = \text{LEARN}(\mathcal{T} + \tilde{\mathcal{P}})$
10 **return** $\mathcal{H}$

[3]Line 9 can be interpreted in different ways, either as $\mathcal{T} \cup \tilde{\mathcal{P}}$ (adopted in this work) or as a weighted combination of LEARN($\mathcal{T}$) and LEARN($\tilde{\mathcal{P}}$) (adopted by [112]).

6.4 Experiments

In Sec. 6.2, we argued from an information theoretic view that ESPA is beneficial for structured tasks if we have a fixed annotation resource. We then proposed SSPAN in Sec. 6.3 to learn from the resulting partial structures. **However**, on one hand, there is still a gap between the I_k analysis and the actual system performance; on the other hand, whether the benefit can be realized in practice also depends on how effective the algorithm exploits partial annotations. Therefore, it remains to be seen how ESPA works in practice. Here we use three NLP tasks: temporal relation (TempRel) extraction, semantic role classification (SRC), and shallow parsing, analogous to the chain, assignment, and BIO structures, respectively.

For all tasks, we compare the following two schemes in Fig. 6.5, where we use graph structures for demonstration. Initially, we have a relatively small but complete dataset $\mathcal{T}_0$, an unannotated dataset $\mathcal{U}_0$, and some budget to annotate $\mathcal{U}_0$. The conventional scheme I, also our baseline here, is to annotate each structure completely before randomly picking up the next one. Due to the limited budget, some $\mathcal{U}_0$ remain untouched (denoted by $\mathcal{U}$). The proposed scheme II adopts ESPA so that all structures at hand are annotated but only partially. For fair comparisons, we use CoDL to incorporate $\mathcal{U}$ into scheme I as well. Finally, the systems trained on the dataset from I/II via CoDL/SSPAN are evaluated on unseen but complete testset $\mathcal{T}_{test}$. Note that because ESPA is a new annotation scheme, there exists no dataset collected this way. We use existing complete datasets and randomly throw out some annotations to mimic ESPA in the following. Due to the randomness in selecting which structures/instances to keep in scheme I/II, we repeat the whole process multiple times and report the mean F_1. The budget, defined as the total number of individual instances that can be annotated, ranges from 10% to 100% with a stepsize of 10%, where x% means x% of all instances in $\mathcal{U}_0$ can be annotated.

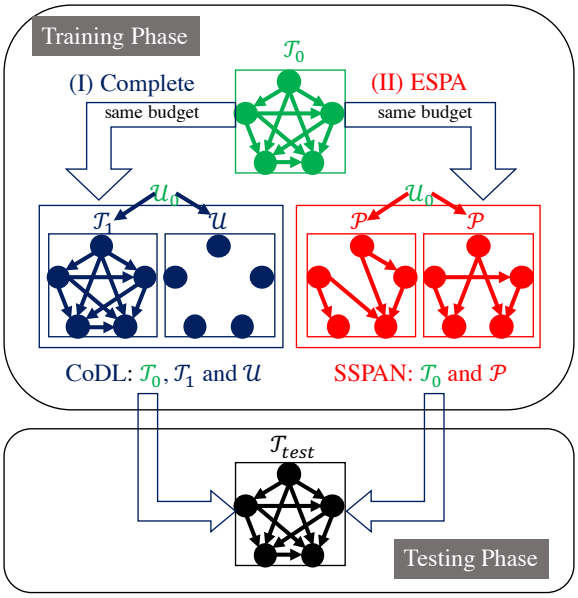
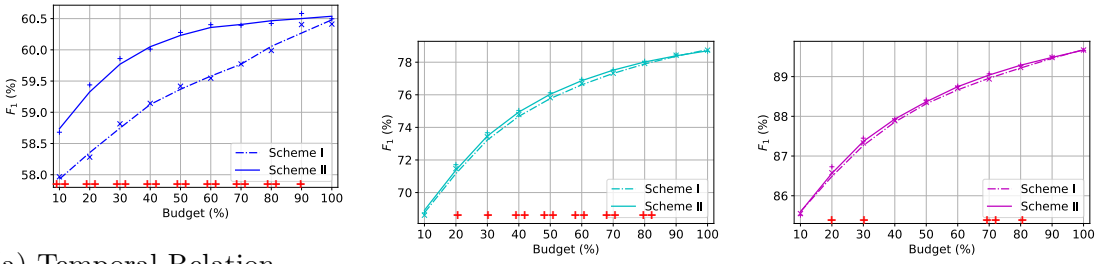


Figure 6.5: The two annotation schemes we compare in Sec. 6.4. $\mathcal{T}$, $\mathcal{P}$, and $\mathcal{U}$ denote complete, partial, and empty structures, respectively. Both schemes start with a complete and relatively small dataset and an unannotated dataset (green). (I) Conventional complete annotation scheme (blue). (II) The proposed ESPA scheme (red). Finally, they are tested on an unseen and complete dataset (black).

6.4.1 Temporal Relation Extraction

Temporal relations (TempRel) are a type of important relations representing the temporal ordering of events described by natural language text. That is to answer questions like which event happens earlier or later in time (see Fig. 6.1a). Since time is physically one-dimensional, if A is before B and B is also before C, then A must be before C. In practice, the label set for TempRels can be more complex, e.g., with labels such as *Simultaneous* and *Vague*, but the structure can still be represented by transitivity constraints (see Table 1 of [49]), which can be viewed as an analogy of the chain structure in Example 23.

To avoid missing relations, annotators are required to exhaustively label every pair of events in a document (i.e., the complete annotation scheme), so it is necessary to study ESPA in this context. Here we adopt the MATRES dataset [12] for its better



(a) Temporal Relation Extraction

(b) SRC

(c) Shallow Parsing

Figure 6.6: Comparison of the baseline, complete annotation scheme and the proposed ESPA scheme (See I & II in Fig. 6.5) under three structured learning tasks (note the scale difference). Each F_1 value is the average of 50 experiments, and each curve is based on corresponding F_1 values smoothed by Savitzky-Golay filters. We can see that scheme II is consistently better than scheme I. Per the Wilcoxon rank-sum test, the significance levels at each given budget are shown on the x-axes, where $+$ and $++$ mean $p < 5\%$ and $p < 1\%$, respectively.

inter-annotator agreement and relatively large size.

Specifically, we use 35 documents as $\mathcal{T}_0$ (the TimeBank-Dense section),[4] 147 documents as $\mathcal{U}_0$ (the TimeBank section minus those documents in $\mathcal{T}_0$), and the Platinum section (a benchmark testset of 20 documents with 1K TempRels) as $\mathcal{T}_{test}$. Note that both schemes I and II are mimicked by downsampling the original annotations in MATRES, where the budget is defined as the total number of TempRels that are kept. Following CogCompTime [153], we choose the same features and sparse-averaged perceptron algorithm as the LEARN component and ILP as INFERENCE for SSPAN.

6.4.2 Semantic Role Classification (SRC)

Semantic role labeling (SRL) is to represent the semantic meanings of language and answer questions like *Who did What to Whom* and *When, Where, How* [34]. Semantic Role Classification (SRC) is a subtask of SRL, which assumes gold predicates and

[4]The original TimeBank-Dense section contains 36 documents, but in collecting MATRES, one of the documents was filtered out because it contained no TempRels between main-axis events.

argument chunks and only classifies the semantic role of each argument. We use the Verb SRL dataset provided by the CoNLL-2005 shared task [206], where the semantic roles include numbered arguments, e.g., *Arg0* and *Arg1*, and argument modifiers, e.g., location (*Am-Loc*), temporal (*Am-Tmp*), and manner (*Am-Mnr*) (see PropBank [37]). The structural constraints for SRC are that each argument can be assigned to exactly one semantic role, and the same role cannot appear twice for a single verb, so SRC is an assignment problem as in Example 24.

Specifically, we use the Wall Street Journal (WSJ) part of Penn TreeBank III [207]. We randomly select 700 sentences from the Sec. 24 of WSJ, among which 100 sentences as $\mathcal{T}_0$ and 600 sentences as $\mathcal{U}_0$. Our $\mathcal{T}_{test}$ is 5700 sentences (about 40K arguments) from Secs. 00, 01, 23. The budget here is defined as the total number of the arguments. We adopt the SRL system in CogCompNLP [208] and use the sparse averaged perceptron as Learn and ILP as Inference.

6.4.3 Shallow Parsing

Shallow parsing, also referred to as *chunking*, is a fundamental NLP task to identify constituents in a sentence, such as noun phrases (NP), verb phrases (VP), and adjective phrases (ADJP), which can be viewed as extending the standard BIO structure in Example 25 with different chunk types: B-NP, I-NP, B-VP, I-VP, B-ADJP, I-ADJP, . . . , O.

We use the chunking dataset provided by the CoNLL-2000 shared task [188]. Specifically, we use 2K tokens' annotations as $\mathcal{T}_0$, 14K tokens as $\mathcal{U}_0$, and the benchmark testset (25K tokens) as $\mathcal{T}_{test}$. The budget here is defined as the total number of tokens' BIO labels. The algorithm we use here is the chunker provided in CogCompNLP, where the Learn component is the sparse averaged perceptron and the Inference is described in [86].

6.4.4 Results

We compare the F_1 performances of all three tasks in Fig. 6.6, averaged over 50 experiments with different randomizations. As the budget increases, the system F_1 increases for both schemes I and II in all three tasks, which confirms the capability of the proposed SSPAN framework to learn from partial structures. When the budget is 100% (i.e., the entire $\mathcal{U}_0$ is annotated), schemes I and II have negligible differences; when the budget is not large enough to cover the entire $\mathcal{U}_0$, scheme II is consistently better than I in all tasks, which follows our expectations based on the I_k analysis. The strict improvement for all budget ratios indicates that the observation is definitely not by chance.

Figure 6.7 further compares the improvement from I to II across tasks. When the budget goes down from 100%, the advantage of ESPA is more prominent; but when the budget is too low, the quality of $\tilde{\mathcal{P}}$ degrades and hurts the performance of SSPAN, leading to roughly hill-shaped curves in Fig. 6.7. We have also conjectured based on Fig. 6.4 that the structure strength goes up from BIO chunks, to bipartite graphs, and to chains; interestingly, the improvement brought by ESPA is consistent with this order.

Admittedly, the improvement, albeit statistically significant, is small, but it does not diminish the contribution of this thesis: Our goal is to remind people that the ESPA scheme (or more generally, partialness) is, at the least, comparable to (or sometimes even better than) complete annotation schemes. Also, the comparison here is in fact unfair to the partial scheme II, because we assume equal cost for both schemes, although it often costs less in a partial scheme as a large problem is decomposed into smaller parts. Therefore, the results shown here imply that the information theoretical benefit of partialness can possibly offset its disadvantages for learning.

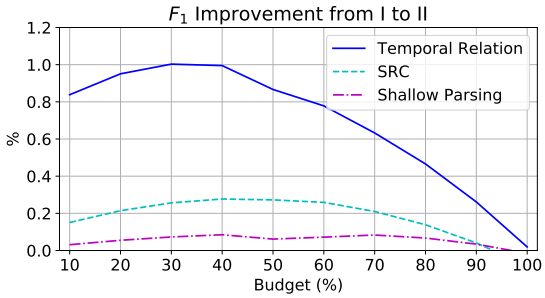


Figure 6.7: The improvement of F_1 brought by ESPA for each task in Fig. 6.6. Note that we conjectured earlier in Fig. 6.4 that the BIO structure is the weakest among the three, which is consistent with the fact that shallow parsing benefits the least from ESPA.

6.5 Discussion

In this chapter, we investigate a less studied, yet important question for structured learning: Given a limited annotation budget (either in time or money), which strategy is better, completely annotating each structure until the budget runs out, or annotating more structures at the cost of leaving some of them partially annotated? Neubig and Mori (2010) [209] investigated this issue specifically in annotating word boundaries and pronunciations for Japanese. Instead of annotating full sentences, they proposed to annotate only some words in a sentence (i.e., partially) that can be chosen heuristically (e.g., skip those that we have seen or those low frequency words). Conceptually, [209] is an active learning work, with the understanding that if the order of annotation is deliberately designed, better learning can be achieved. This thesis addresses the problem from a different angle: Even without active learning, can we still answer the question above?

The observation driving our questions is that when annotating a particular structure, the labels of the yet-to-be-labeled variables may already be constrained by previous annotations and carry less information than those in a totally new structure. Therefore, we systematically study the ESPA scheme – stop annotating a given structure before it is completed and continue annotating another new structure.

An important notion is annotation *cost*. Throughout the chapter we have an ideal assumption that the cost is linear in the total number of annotations, but in practice the case can be more complicated. First, the actual cost of each individual annotation may vary across different instances. We try to eliminate this issue by enforcing random selection of annotation instances, rather than allowing the annotators to select arbitrarily by themselves. This strategy may be useful in practice as well, to avoid people only annotating easy cases. Second, even if we only require labeling partial structures, it is likely that the annotator still needs to comprehend the entire structure, incurring additional cost (usually in terms of time). This issue, however, is not addressed in this work.

Using this definition of cost, we provide a theoretical analysis for ESPA based on the mutual information between target structures and annotation processes. We show that for structures like chains, bipartite graphs, and BIO chunks, the information brought by an extra annotation attenuates as the annotation of the structure is more complete, suggesting to stop early and move to a new structure (although it still remains unclear when it is optimal to stop). This analysis is further supported by experiments on temporal relation extraction, semantic role classification, and shallow parsing, three tasks analogous to the three structures analyzed earlier, respectively. The ratio of the attenuation curve as in Fig. 6.4 is also shown to be an actionable metric to quantify the strength of a type of structure, which can be useful in various analyses, including judging whether ESPA is worthwhile for a particular task. For example, a more detailed I_k-based analysis for SRC shows that predicates with more arguments are stronger structures than those with fewer arguments; we have investigated ESPA on those with more than 6 arguments and indeed observed much larger improvement in SRC. More details on this analysis are in the appendix.

We think that the findings in this chapter are very important. First, as far as we know, we are the first to propose the mutual information analysis that provides a unique view of structured annotation, that of the reduction in the uncertainty of a target of interest $\mathbf{Y}$ by another random variable/process. From this perspective, signals that have non-zero mutual information with $\mathbf{Y}$ can be viewed as "annotations". These can be partially labeled structures (studied here), partial labels (restricting

the possible labels rather than determining a single one as in, e.g., [185]), noisy labels (e.g., generated by crowdsourcing or heuristic rules) or, generally, other indirect supervision signals that are correlated with $\mathbf{Y}$. As we proposed, these can be studied within our mutual information framework as well. This chapter thus provides a way to analyze the benefit of general *incidental supervision signals* [189] and possibly even provides guidance in selecting good incidental supervision signals.

Second, the findings here open up opportunities for new annotation schemes for structured learning. In the past, partially annotated training data have been either a compromise when completeness is infeasible (e.g., when ranking entries in gigantic databases), or collected freely without human annotators (e.g., based on heuristic rules). If we intentionally ask human annotators for partial annotations, the annotation tasks can be more flexible and potentially cost even less. This is because annotating complex structures typically requires certain expertise, and smaller tasks are often easier [203]. It is very likely that some complex annotation tasks require people to read dozens of pages of annotation guidelines, but once decomposed into smaller subtasks, even laymen can handle them. Annotation schemes driven by crowdsourced question-answering, known to provide only partial coverage, are successful examples of this idea [210, 211].

CHAPTER 7

CONCLUSION AND FUTURE WORK

Nowadays, natural language text data exist in enormous amounts and in various forms: books, social media, electronic health records, human-computer dialogues etc. This has created unprecedented opportunities for NLU research. Significant progress has been made in the past two decades, mainly from the token level (e.g., tokenization, lemmatization, part-of-speech tagging, and chunking) and the sentence level (e.g., syntactic parsing, semantic role labeling, and language modeling to some extent). Nevertheless, we still lack a robust solution to various problems that require semantic understandings from the event-level, e.g., *what is going on, what is the cause and impact, and how things will unfold*. With a good understanding of these questions, a computer may be able to interact with humans, read and write stories, predict social, political, or economical trends, and even provide diagnostic guidance based on patient narratives.

This thesis studies one of the core questions in event-level language understanding: *time*. Despite the long-standing philosophical or physical debates over the existence of time, it is undeniably important to have "time" as a special dimension to distinguish things in the past, at present, or in the future, in our everyday life. At the first glance, time may seem to be an easy type of semantics, but it turns out to be a very challenging task and remains unsolved after decades of research. This thesis has contributed to solving this problem in three aspects: time expression understanding, temporal order relation understanding, and temporal common sense understanding.

Chapter 3 focuses on time expression understanding. Time expressions are the most straightforward temporal information that a piece of text can provide to us, and to understand time expressions, we need to chunk them out from text (extraction),

and convert them to a pre-defined format that machines can parse (normalization). We propose to use learning-based methods in the extraction step and rule-based methods in the normalization step. While achieving similar end-to-end performance to state-of-the-art systems, the proposed is much more computationally efficient.

Chapter 4 focuses on temporal order relation understanding (e.g., to determine which event happened earlier or later). We dissect the various structures of time and make use of them in data collection, learning, and inference phases. We investigate the effectiveness of each of them in individual sections and then combine all of them in a unified framework. The state-of-the-art is significantly improved in the proposed method.

Chapter 5 focuses on temporal common sense. Commonsense knowledge is important for computers, but computers usually do not have it because people rarely say the obvious. This chapter investigates this problem in the context of time and proposes five types of temporal common sense. A new dataset is collected via crowdsourcing and used as a testbed to show that existing NLP techniques struggle when it comes to "time."

The three chapters above break down the task of understanding time into three subtasks, all from the angle of information retrieval. We have seen improvements in these subtasks in this thesis. However, the formalism that is used to collect data and evaluate systems is still very rigid, in the sense that it often results in confusing annotations and/or misses many interesting phenomena relevant to time. Recent progress on language modeling (e.g., [149, 166, 177]) has opened up opportunities to annotate natural language tasks using natural language (see [212] for example), and it may be a good direction to bring this topic to another level.

Moreover, we need to clarify that understanding time still requires more than those subtasks. We know that existing machine reading comprehension and textual entailment (which is also called natural language inference) systems easily fail on questions related to time, indicating that these existing NLP techniques do not understand time very well. A more interesting but also more challenging topic is to do *reasoning* on top of temporal information. For instance, to answer questions like *"Does Aristotle have a laptop?"*, the reasoning process requires an understanding of

when Aristotle lived and when the laptop was invented; to make a suggestion to the inquiry *"Where should I go for dinner before the movie tonight?"*, we need to know how long a dinner takes, how long it takes on the road, when the movie starts, and that the dinner should be before the movie starts. It is an important future direction to investigate how to improve machine reading comprehension in terms of time, and on top of that, how to perform temporal reasoning.

In terms of the methodology in NLP, machine learning plays an important role in resolving ambiguity and understanding variability in natural language, including when we tackle those problems above. The standard learning paradigm is to take a specific task, collect data, and train a model for it. The issue is that we will need to have a dedicated dataset for each single task. Since language has high ambiguity and variability, even a small change in the task definition will often require annotating a new dataset. This situation is even more common when we work on events: For temporal relation extraction alone, there are various datasets available, but in most of the cases, we cannot learn from all of them in a single system. We envision that it is crucial for the next generation of machine learning theory to be able to learn from incidental signals, which may be noisy, partial, or only correlated with the task at hand. A deeper understanding of these learning problems will fundamentally transform how we design algorithms and collect data in AI. As an exploration in this direction, Chapter 6 provides a new way to study the partial problem using mutual information. We argue therein that due to the structure of learning problems, only labeling part of a structure also has its benefits: from information theoretic perspective, a finite budget can provide more information if we only annotate structures partially; empirically, we show on several different tasks (including the TempRel extraction task) that partial annotations indeed lead to better learning performance. Still, we want to clarify that Chapter 6 is still a very explorative work and points out future directions towards incidental supervision.

Understanding time may also be helpful for many downstream tasks. For instance, in this information explosion era, it is often important to be able to read and summarize many articles and retrieve the most relevant information. With a timeline, how can we better summarize text? If Event A is almost always followed by Event

B, then from the perspective of information theory, we can remove Event B in our "summary" without distorting the information too much (see a recent work of ours that offers information theoretical understandings of the summarization process of human [213]). Another example is to predict how events will unfold. If machines can really understand temporal information provided in text, then after trawling gigantic data and getting their timelines, we may get important predictive hints for applications in the political, medical, and financial domains.

REFERENCES

[1] M. Verhagen, R. Gaizauskas, F. Schilder, M. Hepple, G. Katz, and J. Pustejovsky, "SemEval-2007 Task 15: TempEval temporal relation identification," in *Proceedings of the 4th International Workshop on Semantic Evaluations*. Association for Computational Linguistics, 2007, pp. 75–80.

[2] M. Verhagen, R. Sauri, T. Caselli, and J. Pustejovsky, "SemEval-2010 Task 13: TempEval-2," in *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 2010, pp. 57–62.

[3] N. UzZaman, H. Llorens, J. Allen, L. Derczynski, M. Verhagen, and J. Pustejovsky, "SemEval-2013 Task 1: TEMPEVAL-3: Evaluating time expressions, events, and temporal relations," *Proceedings of the Joint Conference on Lexical and Computational Semantics (*SEM)*, vol. 2, pp. 1–9, 2013.

[4] J. Strötgen and M. Gertz, "HeidelTime: High quality rule-based extraction and normalization of temporal expressions." in *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 2010, pp. 321–324.

[5] A. X. Chang and C. D. Manning, "SUTIME: A library for recognizing and normalizing time expressions." in *Proceedings of the Language Resources and Evaluation Conference (LREC)*, vol. 2012, 2012, pp. 3735–3740.

[6] R. Zhao, Q. Do, and D. Roth, "A robust shallow temporal reasoning system," in *Proc. of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 6 2012.

[7] N. Chambers, "Event schema induction with a probabilistic entity-driven model." in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, vol. 13, 2013, pp. 1797–1807.

[8] K. Lee, Y. Artzi, J. Dodge, and L. Zettlemoyer, "Context-dependent semantic parsing for time expressions," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2014, pp. 1437–1447.

[9] Q. Ning, Z. Feng, and D. Roth, "A structured learning approach to temporal relation extraction," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Copenhagen, Denmark: Association for Computational Linguistics, 9 2017, pp. 1038–1048.

[10] Q. Ning, Z. Yu, C. Fan, and D. Roth, "Exploiting partially annotated data for temporal relation extraction," in *Proceedings of the Joint Conference on Lexical and Computational Semantics (*SEM)*. Association for Computational Linguistics, 6 2018, pp. 148–153.

[11] Q. Ning, H. Wu, H. Peng, and D. Roth, "Improving temporal relation extraction with a globally acquired statistical resource," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. Association for Computational Linguistics, 6 2018, pp. 841–851.

[12] Q. Ning, H. Wu, and D. Roth, "A multi-axis annotation scheme for event temporal relations," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, 7 2018, pp. 1318–1328.

[13] "The ACE 2005 (ACE 05) Evaluation Plan," Tech. Rep., Sep. 2005.

[14] T. Mitamura, Y. Yamakawa, S. Holm, Z. Song, A. Bies, S. Kulick, and S. Strassel, "Event nugget annotation: Processes and issues," in *Proceedings of the Workshop on Events at NAACL-HLT*, 2015.

[15] Z. Song, A. Bies, S. Strassel, T. Riese, J. Mott, J. Ellis, J. Wright, S. Kulick, N. Ryant, and X. Ma, "From light to rich ERE: Annotation of entities, relations, and events," in *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*. Denver, Colorado: Association for Computational Linguistics, June 2015, pp. 89–98.

[16] H. Ji and R. Grishman, "Refining event extraction through cross-document inference," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2008, pp. 254–262.

[17] S. Liao and R. Grishman, "Using document level cross-event inference to improve event extraction," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2010.

[18] Y. Hong, J. Zhang, B. Ma, J. Yao, G. Zhou, and Q. Zhu, "Using cross-entity inference to improve event extraction," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2011.

[19] R. Huang and E. Riloff, "Bootstrapped training of event extraction classifiers," in *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2012.

[20] R. Huang and E. Riloff, "Modeling textual cohesion for event extraction," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2012.

[21] Q. Li, H. Ji, and L. Huang, "Joint event extraction via structured prediction with global features." in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2013.

[22] B. Yang and T. M. Mitchell, "Joint extraction of events and entities within a document context," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2016.

[23] T. H. Nguyen, K. Cho, and R. Grishman, "Joint event extraction via recurrent neural networks," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. San Diego, California: Association for Computational Linguistics, 2016, pp. 300–309.

[24] C.-Y. Chang, Z. Teng, and Y. Zhang, "Expectation-regulated neural model for event mention extraction," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. San Diego, California: Association for Computational Linguistics, 2016, pp. 400–410.

[25] X. Feng, L. Huang, D. Tang, H. Ji, B. Qin, and T. Liu, "A language-independent neural network for event detection," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Berlin, Germany: Association for Computational Linguistics, 2016, pp. 66–71.

[26] R. Ghaeini, X. Fern, L. Huang, and P. Tadepalli, "Event nugget detection with forward-backward recurrent neural networks," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Berlin, Germany: Association for Computational Linguistics, 2016, pp. 369–373.

[27] O. Tilk, V. Demberg, A. Sayeed, D. Klakow, and S. Thater, "Event participant modelling with neural networks," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Austin, Texas: Association for Computational Linguistics, 2016, pp. 171–182.

[28] T. H. Nguyen and R. Grishman, "Modeling skip-grams for event detection with convolutional neural networks," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

[29] S. Liu, Y. Chen, K. Liu, and J. Zhao, "Exploiting argument information to improve event detection via supervised attention mechanisms," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Vancouver, Canada: Association for Computational Linguistics, 2017, pp. 1789–1798.

[30] W. Orr, P. Tadepalli, and X. Fern, "Event detection with neural networks: A rigorous empirical evaluation," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 999–1004.

[31] H. Peng and D. Roth, "Two discourse driven language models for semantics," in *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[32] H. Peng, Y. Song, and D. Roth, "Event detection and co-reference with minimal supervision," in *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

[33] E. Spiliopoulou, E. Hovy, and T. Mitamura, "Event detection using frame-semantic parser," in *Proceedings of the Events and Stories in the News Workshop*. Vancouver, Canada: Association for Computational Linguistics, August 2017. [Online]. Available: http://www.aclweb.org/anthology/W17-2703 pp. 15–20.

[34] M. Palmer, D. Gildea, and N. Xue, *Semantic Role Labeling*. Morgan & Claypool Publishers, 2010, vol. 3.

[35] V. Punyakanok, D. Roth, and W. Yih, "The importance of syntactic parsing and inference in semantic role labeling," *Computational Linguistics*, vol. 34, no. 2, 2008.

[36] J. Clarke, V. Srikumar, M. Sammons, and D. Roth, "An NLP curator (or: How I learned to stop worrying and love NLP pipelines)," in *Proceedings of the Language Resources and Evaluation Conference (LREC)*, May 2012.

[37] P. Kingsbury and M. Palmer, "From Treebank to PropBank," in *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Spain, 2002.

[38] A. Fader, S. Soderland, and O. Etzioni, "Identifying relations for open information extraction," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2011, pp. 1535–1545.

[39] Mausam, M. Schmitz, R. Bart, S. Soderland, and O. Etzioni, "Open language learning for information extraction," in *Proceedings of Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2012.

[40] G. Angeli, M. J. Johnson Premkumar, and C. D. Manning, "Leveraging linguistic structure for open domain information extraction," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, 2015, pp. 344–354.

[41] A. S. White, D. Reisinger, K. Sakaguchi, T. Vieira, S. Zhang, R. Rudinger, K. Rawlins, and B. Van Durme, "Universal decompositional semantics on universal dependencies," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, 2016, pp. 1713–1723.

[42] Mausam, "Open information extraction systems and downstream applications," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. AAAI Press, 2016, pp. 4074–4077.

[43] Q. Do, Y. S. Chan, and D. Roth, "Minimally supervised event causality identification," in *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Edinburgh, Scotland, July 2011.

[44] Q. Gao, M. Doering, S. Yang, and J. Y. Chai, "Physical causality of action verbs in grounded language understanding." in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[45] C. Hidey and K. McKeown, "Identifying causal relations using parallel Wikipedia articles," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Berlin, Germany: Association for Computational Linguistics, August 2016. [Online]. Available: http://www.aclweb.org/anthology/P16-1135 pp. 1424–1433.

[46] B. Güler, A. Yener, and A. Swami, "Learning causal information flow structures in multi-layer networks," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2016, pp. 1340–1344.

[47] N. Mostafazadeh, A. Grealish, N. Chambers, J. Allen, and L. Vanderwende, "CaTeRS: Causal and temporal relation scheme for semantic annotation of event structures," in *Proceedings of the 4th Workshop on Events: Definition, Detection, Coreference, and Representation*, 2016, pp. 51–61.

[48] P. Mirza and S. Tonelli, "CATENA: CAusal and TEmporal relation extraction from NAtural language texts," in *The 26th International Conference on Computational Linguistics*, 2016, pp. 64–75.

[49] Q. Ning, Z. Feng, H. Wu, and D. Roth, "Joint reasoning for temporal and causal relations," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, 7 2018, pp. 2278–2288.

[50] N. Chambers and D. Jurafsky, "Unsupervised learning of narrative event chains," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2008, pp. 789–797.

[51] N. Chambers and D. Jurafsky, "Unsupervised Learning of Narrative Schemas and their Participants," in *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, ACL 2009*, 2009.

[52] K. Pichotta and R. J. Mooney, "Statistical script learning with multi-argument events," in *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2014.

[53] M. Granroth-Wilding and S. C. Clark, "What happens next? Event prediction using a compositional neural network model," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2016.

[54] Z. Wang, Y. Zhang, and C.-Y. Chang, "Integrating order information and event relation for script event prediction," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

[55] Z. Li, X. Ding, and T. Liu, "Constructing narrative event evolutionary graph for script event prediction," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2018.

[56] K. Humphreys, R. Gaizauskas, and S. Azzam, "Event coreference for information extraction," in *Proceedings of Workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts*, 1997.

[57] A. Bagga and B. Baldwin, "Entity-based cross-document coreferencing using the vector space model," in *Proceedings of the International Conference on Computational Linguistics (COLING)*, 1998.

[58] M. Naughton, "Sentence level event detection and coreference resolution," Ph.D. dissertation, National University of Ireland, Dublin, 2009.

[59] A. Elkhlifi and R. Faiz, "Automatic annotation approach of events in news articles," *International Journal of Computing & Information Sciences*, 2009.

[60] Z. Chen and H. Ji, "Graph-based event coreference resolution," in *Proceedings of the Workshop on Graph-based Methods for Natural Language Processing*, 2009.

[61] L. Huang, T. Cassidy, X. Feng, H. Ji, C. R. Voss, J. Han, and A. Sil, "Liberal event extraction and event schema induction," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[62] S. Pradhan, L. Ramshaw, R. Weischedel, J. MacBride, and L. Micciulla, "Unrestricted coreference: Identifying entities and events in ontonotes," in *ICSC*, 2007.

[63] H. Lee, M. Recasens, A. Chang, M. Surdeanu, and D. Jurafsky, "Joint entity and event coreference resolution across documents," in *Proceedings of the SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, 2012.

[64] J. Lu and V. Ng, "Joint learning for event coreference resolution," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[65] C. Bejan and S. Harabagiu, "Unsupervised event coreference resolution with rich linguistic features," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2010.

[66] T. Zhang, H. Li, H. Ji, and S.-F. Chang, "Cross-document event coreference resolution based on cross-media features," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

[67] S. Barhom, V. Shwartz, A. Eirew, M. Bugert, N. Reimers, and I. Dagan, "Revisiting joint modeling of cross-document entity and event coreference resolution," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[68] N. Burton-Roberts, *Analysing Sentences: An Introduction to English Syntax*, 4th ed. London and New York: Routledge, Taylor & Francis Group, 2016, ch. 1, pp. 6–10.

[69] J. D. Gorman and A. O. Hero, "Lower bounds for parametric estimation with constraints," *IEEE Transactions on Information Theory*, vol. 36, no. 6, pp. 1285–1301, 1990.

[70] M. Held and R. M. Karp, "The traveling-salesman problem and minimum spanning trees: Part II," *Mathematical Programming*, vol. 1, no. 1, pp. 6–25, 1971.

[71] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo (Calif.), 1988.

[72] A. M. Rush, D. Sontag, M. Collins, and T. Jaakkola, "On dual decomposition and linear programming relaxations for natural language processing," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2010.

[73] A. M. Rush and M. Collins, "A tutorial on dual decomposition and lagrangian relaxation for inference in natural language processing," *Journal of Artificial Intelligence Research*, vol. 45, pp. 305–362, 2012.

[74] D. Roth and W. Yih, "A linear programming formulation for global inference in natural language tasks," in *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*, H. T. Ng and E. Riloff, Eds. Association for Computational Linguistics, 2004, pp. 1–8.

[75] Gurobi Optimization, Inc., "Gurobi optimizer reference manual," 2015.

[76] M. Padberg and G. Rinaldi, "A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems," *SIAM Review*, vol. 33, no. 1, pp. 60–100, 1991.

[77] V. Srikumar, G. Kundu, and D. Roth, "On amortizing inference cost for structured prediction," in *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 7 2012.

[78] G. Kundu, V. Srikumar, and D. Roth, "Margin-based decomposed amortized inference," in *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 8 2013.

[79] K.-W. Chang, S. Upadhyay, G. Kundu, and D. Roth, "Structural learning with amortized inference," in *Proc. of the Conference on Artificial Intelligence (AAAI)*, 2015.

[80] V. Punyakanok, D. Roth, W. Yih, and D. Zimak, "Learning and inference over constrained output," in *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2005, pp. 1124–1129.

[81] G. Angeli, C. Manning, and D. Jurafsky, "Parsing time: Learning to interpret time expressions," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, June 2012.

[82] E. Laparra, D. Xu, A. Elsayed, S. Bethard, and M. Palmer, "SemEval 2018 task 6: Parsing time normalizations," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, 2018, pp. 88–96.

[83] S. Bethard and J. Parker, "A semantically compositional annotation scheme for time normalization," in *Proceedings of the Language Resources and Evaluation Conference (LREC)*. Portorož, Slovenia: European Language Resources Association (ELRA), 2016, pp. 3779–3786.

[84] E. Laparra, D. Xu, and S. Bethard, "From characters to time intervals: New paradigms for evaluation and neural parsing of time normalizations," *Transactions of the Association for Computational Linguistics (TACL)*, vol. 6, pp. 343–356, 2018.

[85] D. Xu, E. Laparra, and S. Bethard, "Pre-trained contextualized character embeddings lead to major improvements in time normalization: a detailed analysis," in *Proceedings of the Joint Conference on Lexical and Computational Semantics (*SEM)*, 2019, pp. 68–74.

[86] V. Punyakanok and D. Roth, "The use of classifiers in sequential inference," in *Proc. of the Conference on Neural Information Processing Systems (NIPS)*. MIT Press, 2001, pp. 995–1001.

[87] K. Vijay-Shanker and D. J. Weir, "Polynomial time parsing of combinatory categorial grammars," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 1990.

[88] M. Kuhlmann, G. Satta, and P. Jonsson, "On the complexity of CCG parsing," *Computational Linguistics*, vol. 44, no. 3, 2018.

[89] J. Pustejovsky, P. Hanks, R. Sauri, A. See, R. Gaizauskas, A. Setzer, D. Radev, B. Sundheim, D. Day, L. Ferro et al., "The TIMEBANK corpus," in *Corpus Linguistics*, 2003, p. 40.

[90] J. F. Allen, "Towards a general theory of action and time," *Artificial Intelligence*, vol. 23, no. 2, pp. 123–154, 1984.

[91] S. Bethard, J. H. Martin, and S. Klingenstein, "Timelines from text: Identification of syntactic temporal relations," in *Semantic Computing, 2007. ICSC 2007. International Conference on*. IEEE, 2007, pp. 11–18.

[92] Q. Do, W. Lu, and D. Roth, "Joint inference for event timeline construction," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2012.

[93] T. Cassidy, B. McDowell, N. Chambers, and S. Bethard, "An annotation framework for dense event ordering," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2014, pp. 501–506.

[94] I. Mani, M. Verhagen, B. Wellner, C. M. Lee, and J. Pustejovsky, "Machine learning of temporal relations," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2006, pp. 753–760.

[95] N. Chambers, T. Cassidy, B. McDowell, and S. Bethard, "Dense event ordering with a multi-pass architecture," *Transactions of the Association for Computational Linguistics (TACL)*, vol. 2, pp. 273–284, 2014.

[96] N. Chambers, S. Wang, and D. Jurafsky, "Classifying temporal relations between events," in *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics, 2007, pp. 173–176.

[97] M. Verhagen and J. Pustejovsky, "Temporal processing with the TARSQI toolkit," in *22nd International Conference on Computational Linguistics: Demonstration Papers*. Association for Computational Linguistics, 2008, pp. 189–192.

[98] S. Bethard, "ClearTK-TimeML: A minimalist approach to TempEval 2013," in *Proceedings of the Joint Conference on Lexical and Computational Semantics (*SEM)*, vol. 2, 2013, pp. 10–14.

[99] N. Laokulrat, M. Miwa, Y. Tsuruoka, and T. Chikayama, "UTTime: Temporal relation classification using deep syntactic features," in *Second Joint Conference on Lexical and Computational Semantics (* SEM)*, vol. 2, 2013, pp. 88–92.

[100] N. Chambers, "NavyTime: Event and time ordering from raw text." DTIC Document, Tech. Rep., 2013.

[101] P. Bramsen, P. Deshpande, Y. K. Lee, and R. Barzilay, "Inducing temporal graphs," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Sydney, Australia: Association for Computational Linguistics, July 2006. [Online]. Available: http://www.aclweb.org/anthology/W/W06/W06-1623 pp. 189–198.

[102] I. Mani, B. Wellner, M. Verhagen, and J. Pustejovsky, "Three approaches to learning TLINKs in TimeML," *Technical Report CS-07–268, Computer Science Department*, 2007.

[103] T. Leeuwenberg and M.-F. Moens, "Structured learning for temporal relation extraction from clinical records," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 2017.

[104] S. Bethard, L. Derczynski, G. Savova, J. Pustejovsky, and M. Verhagen, "SemEval-2015 Task 6: Clinical TempEval," in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. Denver, Colorado: Association for Computational Linguistics, June 2015. [Online]. Available: http://www.aclweb.org/anthology/S15-2136 pp. 806–814.

[105] S. Bethard, G. Savova, W.-T. Chen, L. Derczynski, J. Pustejovsky, and M. Verhagen, "SemEval-2016 Task 12: Clinical TempEval," in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. San Diego, California: Association for Computational Linguistics, June 2016. [Online]. Available: http://www.aclweb.org/anthology/S16-1165 pp. 1052–1062.

[106] S. Bethard, G. Savova, M. Palmer, and J. Pustejovsky, "Semeval-2017 task 12: Clinical tempeval," in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, 2017, pp. 565–572.

[107] Y. Zhang and N. Xue, "Structured interpretation of temporal relations," *Proceedings of the Language Resources and Evaluation Conference (LREC)*, 2018.

[108] Y. Zhang and N. Xue, "Acquiring structured temporal representation via crowdsourcing: A feasibility study," in *Proceedings of the Joint Conference on Lexical and Computational Semantics (*SEM)*. Association for Computational Linguistics, 2019, pp. 178–185.

[109] Y. Zhang and N. Xue, "Neural ranking models for temporal dependency structure parsing," *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[110] M. Collins, "Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2002.

[111] M.-W. Chang, V. Srikumar, D. Goldwasser, and D. Roth, "Structured output learning with indirect supervision," in *Proc. of the International Conference on Machine Learning (ICML)*, 2010.

[112] M. Chang, L. Ratinov, and D. Roth, "Guiding semi-supervision with constraint-driven learning," in *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Prague, Czech Republic: Association for Computational Linguistics, 6 2007, pp. 280–287.

[113] M.-W. Chang, L. Ratinov, and D. Roth, "Structured learning with constrained conditional models," *Machine Learning*, vol. 88, no. 3, pp. 399–431, 6 2012.

[114] Y. Freund and R. Schapire, "Large margin classification using the Perceptron algorithm," in *Proceedings of the Annual ACM Workshop on Computational Learning Theory (COLT)*, 1998, pp. 209–217.

[115] D. Graff, "The AQUAINT corpus of English news text," *Linguistic Data Consortium*, 2002.

[116] N. Rizzolo and D. Roth, "Learning based Java for rapid development of NLP systems," in *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Valletta, Malta, May 2010.

[117] B. S. Everitt, *The Analysis of Contingency Tables*. CRC Press, 1992.

[118] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Computation*, vol. 10, no. 7, pp. 1895–1924, 1998.

[119] T. Jiang, T. Liu, T. Ge, L. Sha, B. Chang, S. Li, and Z. Sui, "Towards time-aware knowledge graph completion," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan: The COLING 2016 Organizing Committee, December 2016. [Online]. Available: http://aclweb.org/anthology/C16-1161 pp. 1715–1724.

[120] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum, "YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia (extended abstract)," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2013.

[121] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*. ACM, 2008, pp. 1247–1250.

[122] T. Chklovski and P. Pantel, "VerbOcean: Mining the web for fine-grained semantic verb relations," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2004, pp. 33–40.

[123] E. Hovy, T. Mitamura, F. Verdejo, J. Araki, and A. Philpot, "Events are not simple: Identity, non-identity, and quasi-identity," in *Workshop on Events*, 2013.

[124] P. Denis and P. Muller, "Predicting globally-coherent temporal structures from texts via endpoint inference and graph decomposition," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 22, no. 3, 2011, p. 1788.

[125] T. O'Gorman, K. Wright-Bettner, and M. Palmer, "Richer event description: Integrating event coreference with temporal, causal and bridging annotation," in *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*. Austin, Texas: Association for Computational Linguistics, November 2016, pp. 47–56.

[126] W. F. Styler IV, S. Bethard, S. Finan, M. Palmer, S. Pradhan, P. C. de Groen, B. Erickson, T. Miller, C. Lin, G. Savova et al., "Temporal annotation in the clinical domain," *Transactions of the Association for Computational Linguistics (TACL)*, vol. 2, p. 143, 2014.

[127] M. Coll-Florit and S. P. Gennari, "Time in language: Event duration in language comprehension," *Cognitive Psychology*, vol. 62, no. 1, pp. 41–79, 2011.

[128] J. Pustejovsky, J. Castao, R. Ingria, R. Saur, R. Gaizauskas, A. Setzer, and G. Katz, "TimeML: Robust specification of event and temporal expressions in text," in *in Fifth International Workshop on Computational Semantics (IWCS-5)*, 2003.

[129] S. Bethard, O. Kolomiyets, and M.-F. Moens, "Annotating story timelines as temporal dependency structures," in *Proceedings of the Language Resources and Evaluation Conference (LREC)*. ELRA, 2012, pp. 2721–2726.

[130] R. Saurí and J. Pustejovsky, "FactBank: a corpus annotated with event factuality," *Proceedings of the Language Resources and Evaluation Conference (LREC)*, vol. 43, no. 3, p. 227, 2009.

[131] K. Lee, Y. Artzi, Y. Choi, and L. Zettlemoyer, "Event detection and factuality assessment with non-expert supervision," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Lisbon, Portugal: Association for Computational Linguistics, September 2015, pp. 1643–1648.

[132] P. Raghavan, E. Fosler-Lussier, and A. M. Lai, "Learning to temporally order medical events in clinical text," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, 2012, pp. 70–74.

[133] S. Schockaert and M. De Cock, "Temporal reasoning about fuzzy intervals," *Artificial Intelligence*, vol. 172, no. 8-9, pp. 1158–1193, 2008.

[134] C. Fellbaum, *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[135] D. Zeng, K. Liu, S. Lai, G. Zhou, and J. Zhao, "Relation classification via convolutional deep neural network," in *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2014, pp. 2335–2344.

[136] S. Zhang, D. Zheng, X. Hu, and M. Yang, "Bidirectional long short-term memory networks for relation classification," in *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, 2015, pp. 73–78.

[137] D. Zhang and D. Wang, "Relation classification via recurrent neural network," *arXiv preprint arXiv:1508.01006*, 2015.

[138] Z. Xu, J. Hu, and W. Deng, "Recurrent convolutional neural network for video classification," in *Multimedia and Expo (ICME), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–6.

[139] D. Dligach, T. Miller, C. Lin, S. Bethard, and G. Savova, "Neural temporal relation extraction," in *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, vol. 2, 2017, pp. 746–751.

[140] C. Lin, T. Miller, D. Dligach, S. Bethard, and G. Savova, "Representations of time expressions for temporal relation extraction with convolutional neural networks," *BioNLP 2017*, pp. 322–327, 2017.

[141] J. Tourille, O. Ferret, A. Neveol, and X. Tannier, "Neural architecture for temporal relation extraction: A Bi-LSTM approach for detecting narrative containers," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, vol. 2, 2017, pp. 224–230.

[142] F. Cheng and Y. Miyao, "Classifying temporal relations by bidirectional LSTM over dependency paths," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, vol. 2, 2017, pp. 1–6.

[143] Y. Meng and A. Rumshisky, "Context-aware neural model for temporal information extraction," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, vol. 1, 2018, pp. 527–536.

[144] A. Leeuwenberg and M.-F. Moens, "Temporal information extraction by predicting relative time-lines," *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[145] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[146] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in *Proceedings of NIPS 2013*, 2013, pp. 1–9.

[147] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014, pp. 1532–1543.

[148] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[149] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018.

[150] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

[151] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," in *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 1994, pp. 737–744.

[152] A. Patel, A. Sands, C. Callison-Burch, and M. Apidianaki, "Magnitude: A fast, efficient universal vector embedding utility package," *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[153] Q. Ning, B. Zhou, Z. Feng, H. Peng, and D. Roth, "Cogcomptime: A tool for understanding time in natural language," in *EMNLP (Demo Track)*. Brussels, Belgium: Association for Computational Linguistics, 11 2018.

[154] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[155] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Computation*, 1998.

[156] L. Schubert, "Can we derive general world knowledge from texts?" in *Proceedings of the second international conference on Human Language Technology Research*. Morgan Kaufmann Publishers Inc., 2002, pp. 94–97.

[157] E. Davis, *Representations of Commonsense Knowledge*. Morgan Kaufmann, 2014.

[158] S. Zhang, R. Rudinger, K. Duh, and B. Van Durme, "Ordinal commonsense inference," *Transactions of the Association for Computational Linguistics (TACL)*, vol. 5, no. 1, pp. 379–395, 2017.

[159] L. Bauer, Y. Wang, and M. Bansal, "Commonsense for generative multi-hop question answering tasks," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018, pp. 4220–4230.

[160] N. Tandon, B. Dalvi, J. Grus, W.-t. Yih, A. Bosselut, and P. Clark, "Reasoning about actions and state changes by injecting commonsense knowledge," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018, pp. 57–66.

[161] A. Gusev, N. Chambers, P. Khaitan, D. Khilnani, S. Bethard, and D. Jurafsky, "Using query patterns to learn the duration of events," in *IWCS*. Association for Computational Linguistics, 2011, pp. 145–154.

[162] J. Williams, "Extracting fine-grained durations for verbs from twitter," in *Proceedings of ACL 2012 Student Research Workshop*. Association for Computational Linguistics, 2012, pp. 49–54.

[163] Z. Kozareva and E. Hovy, "Learning temporal information for states and events," in *Fifth International Conference on Semantic Computing*. IEEE, 2011, pp. 424–429.

[164] M. Roemmele, C. A. Bejan, and A. S. Gordon, "Choice of plausible alternatives: An evaluation of commonsense causal reasoning," in *2011 AAAI Spring Symposium Series*, 2011.

[165] Q. Chen, X. Zhu, Z. Ling, S. Wei, H. Jiang, and D. Inkpen, "Enhanced LSTM for natural language inference," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Vancouver: ACL, July 2017.

[166] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

[167] M. Forbes and Y. Choi, "Verb physics: Relative physical knowledge of actions and objects," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, vol. 1, 2017, pp. 266–276.

[168] Y. Yang, L. Birnbaum, J.-P. Wang, and D. Downey, "Extracting commonsense properties from embeddings with limited human guidance," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, vol. 2, 2018, pp. 644–649.

[169] A. Cocos, V. Wharton, E. Pavlick, M. Apidianaki, and C. Callison-Burch, "Learning scalar adjective intensity from paraphrases," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018, pp. 1752–1762.

[170] H. Rashkin, M. Sap, E. Allaway, N. A. Smith, and Y. Choi, "Event2Mind: Commonsense inference on events, intents, and reactions," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018, pp. 463–473.

[171] R. Zellers, Y. Bisk, R. Schwartz, and Y. Choi, "SWAG: A large-scale adversarial dataset for grounded commonsense inference," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018, pp. 93–104.

[172] A. Vempala, E. Blanco, and A. Palmer, "Determining event durations: Models and error analysis," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, vol. 2, 2018, pp. 164–168.

[173] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, "Think you have solved question answering? Try ARC, the AI2 reasoning challenge," *arXiv preprint arXiv:1803.05457*, 2018.

[174] S. Ostermann, M. Roth, A. Modi, S. Thater, and M. Pinkal, "Semeval-2018 task 11: Machine comprehension using commonsense knowledge," in *SemEval*, 2018, pp. 747–757.

[175] E. Merkhofer, J. Henderson, D. Bloom, L. Strickhart, and G. Zarrella, "Mitre at semeval-2018 task 11: Commonsense reasoning without commonsense knowledge," in *SemEval*, 2018, pp. 1078–1082.

[176] D. Khashabi, S. Chaturvedi, M. Roth, S. Upadhyay, and D. Roth, "Looking beyond the surface: A challenge set for reading comprehension over multiple sentences," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018.

[177] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[178] A. K. A. F. Jonghyun Choi, Jayant Krishnamurthy, "Structured set matching networks for one-shot part labeling," *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[179] D. Angluin, "Queries and concept learning," *Machine Learning*, vol. 2, no. 4, pp. 319–342, April 1988.

[180] L. E. Atlas, D. A. Cohn, and R. E. Ladner, "Training connectionist networks with queries and selective sampling," in *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 1990, pp. 566–573.

[181] D. Lewis and W. Gale, "Training text classifiers by uncertainty sampling," in *Proceedings of ACM-SIGIR Conference on Information Retrieval*, 1994.

[182] D. Roth and K. Small, "Active learning with perceptron for structured output," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2006.

[183] D. Roth and K. Small, "Margin-based active learning for structured output spaces," in *Proc. of the European Conference on Machine Learning (ECML)*. Springer, 9 2006.

[184] D. Roth and K. Small, "Active learning for pipeline models," in *Proc. of the Conference on Artificial Intelligence (AAAI)*, 7 2008.

[185] P. Hu, Z. Lipton, A. Anandkumar, and D. Ramanan, "Active learning with partial feedback," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[186] V. Ambati, S. Vogel, and J. G. Carbonell, "Active learning and crowd-sourcing for machine translation." in *Proceedings of the Language Resources and Evaluation Conference (LREC)*, vol. 1. Citeseer, 2010, p. 2.

[187] F. Laws, C. Scheible, and H. Schütze, "Active learning with Amazon Mechanical Turk," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2011, pp. 1546–1556.

[188] E. F. Tjong Kim Sang and S. Buchholz, "Introduction to the CoNLL-2000 shared task: Chunking," in *Proceedings of the CoNLL-2000 and LLL-2000*, 2000, pp. 127–132.

[189] D. Roth, "Incidental supervision: Moving beyond supervised learning," in *Proc. of the Conference on Artificial Intelligence (AAAI)*, 2 2017.

[190] G. Brightwell and P. Winkler, "Counting linear extensions is #p-complete," in *Proceedings of the Twenty-third Annual ACM Symposium on Theory of Computing*, 1991, pp. 175–181.

[191] O. Shamir, S. Sabato, and N. Tishby, "Learning and generalization with the information bottleneck," *Theoretical Computer Science*, vol. 411, no. 29-30, pp. 2696–2711, 2010.

[192] R. Shwartz-Ziv and N. Tishby, "Opening the black box of deep neural networks via information," *arXiv preprint arXiv:1703.00810*, 2017.

[193] S. Yu and J. C. Principe, "Understanding autoencoders with information theoretic concepts," *arXiv preprint arXiv:1804.00057*, 2018.

[194] A. M. Saxe, Y. Bansal, J. Dapello, M. Advani, A. Kolchinsky, B. D. Tracey, and D. D. Cox, "On the information bottleneck theory of deep learning," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[195] H. Scudder, "Probability of error of some adaptive pattern-recognition machines," *IEEE Transactions on Information Theory*, vol. 11, no. 3, pp. 363–371, 1965.

[196] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervied methods," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 1995.

[197] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the Annual ACM Workshop on Computational Learning Theory (COLT)*, 1998, pp. 92–100.

[198] K. Nigam, A. K. Mccallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," *Machine Learning*, vol. 39, no. 2/3, pp. 103–134, 2000.

[199] T. Joachims, "Transductive inference for text classification using support vector machines," in *Proceedings of the International Conference on Machine Learning (ICML)*, 1999.

[200] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society, Series B*, vol. 39, no. 1, pp. 1–38, 1977.

[201] K. Ganchev, J. Graça, J. Gillenwater, and B. Taskar, "Posterior regularization for structured latent variable models," *Journal of Machine Learning Research*, 2010.

[202] Y. Tsuboi, H. Kashima, H. Oda, S. Mori, and Y. Matsumoto, "Training conditional random fields using incomplete annotations," in *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2008, pp. 897–904.

[203] E. R. Fernandes and U. Brefeld, "Learning from partially annotated sequences," in *Proceedings of the Joint European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2011.

[204] D. Hovy and E. Hovy, "Exploiting partial annotations with EM training," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2012, pp. 31–38.

[205] X. Lou and F. A. Hamprecht, "Structured learning from partial annotations," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2012, pp. 371–378.

[206] X. Carreras and L. Màrquez, "Introduction to the CoNLL-2005 shared task: Semantic role labeling," in *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*. Ann Arbor, Michigan: Association for Computational Linguistics, June 2005. [Online]. Available: http://www.aclweb.org/anthology/W/W05/W05-0620 pp. 152–164.

[207] M. P. Marcus, B. Santorini, and M. Marcinkiewicz, "Building a large annotated corpus of English: The Penn Treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, June 1993.

[208] D. Khashabi, M. Sammons, B. Zhou, T. Redman, C. Christodoulopoulos, V. Srikumar, N. Rizzolo, L. Ratinov, G. Luo, Q. Do, C.-T. Tsai, S. Roy, S. Mayhew, Z. Feng, J. Wieting, X. Yu, Y. Song, S. Gupta, S. Upadhyay, N. Arivazhagan, Q. Ning, S. Ling, and D. Roth, "CogCompNLP: Your Swiss Army Knife for NLP," in *Proceedings of the Language Resources and Evaluation Conference (LREC)*, 2018.

[209] G. Neubig and S. Mori, "Word-based partial annotation for efficient corpus construction," in *Proceedings of the Language Resources and Evaluation Conference (LREC)*. Citeseer, 2010.

[210] L. He, M. Lewis, and L. Zettlemoyer, "Question-answer driven semantic role labeling: Using natural language to annotate natural language," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015, pp. 643–653.

[211] J. Michael, G. Stanovsky, L. He, I. Dagan, and L. Zettlemoyer, "Crowdsourcing question-answer meaning representations," *arXiv preprint arXiv:1711.05885*, 2017.

[212] H. He, Q. Ning, and D. Roth, "Incidental supervision from question-answering signals," *arXiv preprint arXiv:1909.00333*, 2019.

[213] E. Graves, Q. Ning, and P. Basu, "An information theoretic model for summarization, and some basic results," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2019.