博士論文

Exploiting Non-Local Information in Relation Extraction from Documents

(文書からの関係抽出における非局所的情報の利用)

ラオクンラット　ナッスダ

ACKNOWLEDGEMENTS

In the first place, I would like to record my sincerest gratitude to my supervisor, Associate Professor Yoshimasa Tsuruoka, for his supervision and supporting me throughout this thesis. This work will never be completed without his advice and guidance.

Secondly, I would like to thank my former supervisor, Professor Masanori Sugimoto, who gave me the best chance in my life to come to Japan and pursue my graduate study at the University of Tokyo.

I gratefully thank Professor Takashi Chikayama and Associate Professor Makato Miwa, who always give me good advice and help. Technical discussions with them and their invaluable insight always helped me improve my work.

My special thanks go to Professor Hiromichi Hashizume, Yasushige Maeda, and every member of Interaction Technology Laboratory, especially Natapon Pantuwong, Thitirat Siriborvornratanakul, and Nuttapol Sangsuriyachot.

I gratefully acknowledge everyone in Chikayama-Tsuruoka Laboratory, especially Kazuma Hashimoto and Nguyen Thi Hong Nhung, who have helped me in many things.

Finally, I would like to thank my family who always support me in every situation. I am sincerely grateful to them for their understanding and encouragement during my time in Japan.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

LIST OF ABBREVIATIONS

NLP Natural Language Processing

POS Part of Speech

CRF Conditional Random Field

SVM Support Vector Machine

ADJ Adjacent nodes and links

OP Other paths

GP Generalized paths

EVE (E,V,E) tuples

VEV (V,E,V) tuples

MLN Markov Logic Network

ILP Integer Linear Programming

ABSTRACT

Exploiting Non-Local Information in Relation Extraction from Documents
by
Natsuda Laokulrat

Supervisor: Yoshimasa Tsuruoka

Relation extraction from documents is one of the most common tasks in Natural Language Processing (NLP). Relations between entities are an important piece of information for deep understanding of documents. Moreover, being able to extract relations appearing in text is beneficial for various NLP applications such as textual entailment, multi-document summarization, and question answering.

Traditional machine learning-based approaches to relation extraction use only local features, i.e., features between a given pair of entities, and thus fail to incorporate useful information that could be inferred from nearby entities into the classification process. In this project, in order to apply non-local information into the classification, we make use of graphs and apply a stacked learning method to classification task.

This study mainly focuses on the temporal relation classification. However, the idea can also be applied to other NLP relation extraction tasks. In this work, the temporal relation classification task has been performed with extensive experiments to verify the proposed method.

Temporal relation classification aims to classify temporal relationships between pairs of temporal entities into one of the relation types such as BEFORE, AFTER, SIMULTANEOUS, and BEGINS. Local approaches do not consider entities that have temporal connections to the entities in the given pair at all, and thus contradictions within a document can occur. For instance, the system may predict that A happens before B, that B happens before C, and that A happens after C, which are mutually contradictory. In our model, we tackle the problem of contradictory predictions by using a stacked learning approach. The prediction for a temporal relation is made by considering the consistency of possible relations between nearby entities.

In this study, we tackle the problem of contradictory predictions by using a *stacked learning* approach proposed by *Wolpert* (1992). Stacked learning is a machine learning framework that allows one to incorporate non-local information into a structured prediction problem and has proven useful in dependency parsing (*Martins et al.* (2008)). We employ stacked learning in order to use the results of *temporal inference* as non-local features in temporal relation classification. To perform temporal inference, we use timegraphs proposed by *Miller and Schubert* (1990), which represent temporal connectivity of all temporal entities in each document.

Global approaches for tackling the aforementioned problem have been proposed previously (*Chambers and Jurafsky* (2008); *Yoshikawa et al.* (2009); *Denis and Muller* (2011); *Do et al.* (2012)). Chambers and Jurafsky used Integer Linear Programming (ILP) to maximize the confidence scores of the output of local classifiers in order to solve the contradictory prediction. Denis and Muller also used ILP but they enforced temporal relation coherence only on particular sets of events rather than on the entire documents. However, both of the studies focused only on the temporal relations between events and used reduced sets of the temporal relations, i.e., Chambers and Jurafsky used *BEFORE*, *AFTER*, and *VAGUE.*, while Denis and Muller used *BEFORE*, *AFTER*, *OVERLAP*, and *NO RELATION.* Do et al. employed a full set of temporal relations to construct a globally coherent timeline for an article using ILP, leveraged event coreference to support timeline construction, and associated each event with a precise time interval.

Yoshikawa et al. proposed a Markov Logic model to jointly predict the temporal relations between events and time expressions. They also used a reduced set of the relation types, i.e., *BEFORE*, *OVERLAP*, *AFTER*, *BEFORE-OR-OVERLAP*, *OVERLAP-OR-AFTER*, and *VAGUE*.

Our method differs from theirs in that their methods used transition rules to enforce consistency within each triplet of relations, but our method can also work with a set consisting of more than three relations. Moreover, in our work, the full set of temporal relations specified in TimeML are used, rather than the reduced set used in *Chambers and Jurafsky* (2008) and *Yoshikawa et al.* (2009).

We evaluate our method on the TempEval-3's Task C-relation-only data, which provides a system with all the appropriate temporal links and only needs the system to classify the relation types. The results show that by exploiting the probability values in the stacked learning approach, the classification performance improves significantly. By performing 10-fold cross validation on the Timebank corpus, we can achieve an F1 score of 60.25% based on the graph-based evaluation, which is 0.90 percentage points *(pp)* higher than that of the local approach.

We compared our system to the state-of-the-art systems that use global information in temporal relation classification and found that our system outperforms those systems. Our system can achieve 7.7 *pp* higher accuracy than Chambers's system and 0.9 *pp* higher accuracy than Yoshikawa's system. By using a stacked learning approach, we are able to include a large number of features into our models, which makes our results better than those of *Yoshikawa et al.* (2009), since including a large number of features into a Markov Logic model is difficult and computationally expensive.

I Introduction

1.1 Natural Language Processing

Natural Language Processing (NLP) is becoming more and more important in this digital era when information is everywhere, becoming too large and growing too rapidly. NLP has been widely used and already become a part of our daily lives. The most common applications of NLP are search engines, smart phones' autocorrection function, and machine translation. More advanced applications have been invented by giant companies, such as Apple's Siri, Google Now, and IBM Watson.

Apple's Siri, an application for Apple's iOS, uses a natural language user interface to answer questions, make recommendations, and perform actions by delegating requests to a set of Web services. Fig.1.1 demonstrates how Siri answers the questions.

IBM Watson is an artificial intelligence computer system capable of answering questions posed in natural language, developed in IBM's DeepQA project (`www-03.ibm.com/innovation/us/watson/index.html`). In 2011, Watson competed on the

Figure 1.1: Apple's Siri

Figure 1.2: IBM Watson

Figure 1.3: Google Search

quiz show Jeopardy!, as shown in Fig.1.2, and received the first prize. It had access to 200 million pages of structured and unstructured content consuming four terabytes of disk storage, including the full text of Wikipedia, but it was not connected to the Internet during the game.

Google Search, as shown in Fig.1.3, is a very success application of NLP and has become an essential part of our daily lives. Another state-of-the-art NLP system is WolframAlpha (`www.wolframalpha.com`), developed by Wolfram Research. It can give answers on facts and data and calculates answers across a range of topics, including science, nutrition, history, geography, engineering, mathematics, linguistics, sports, etc.

1.2 Relation extraction

Normal text documents are unstructured. In order to change unstructured information into computer-understandable data, a task called relation extraction is necessary. Relation extraction from documents is one of the most common tasks in NLP. There exist many kinds of relation that can be extracted from text, such as dependencies between word tokens (dependency parsing), temporal relation between entities within or across documents.

Figure 1.4: Example sentence for temporal relation classification

Relations between entities are an important piece of information for deep understanding of documents. Moreover, being able to extract relations appearing in text is beneficial for various NLP applications such as textual entailment, multi-document summarization, and question answering.

Traditional machine learning-based approaches to relation extraction use only local features, i.e., features between a given pair of entities, and thus fail to incorporate useful information that could be inferred from nearby entities into the classification process.

Temporal relation classification aims to classify temporal relationships between pairs of temporal entities into one of the relation types such as BEFORE, AFTER, SIMULTANEOUS, and BEGINS.

Event extraction is a task to identify and classify events in the document text, as well as arguments associated with those events. For classification tasks, the traditional approach is to use local information associated with trigger and argument tokens.

1.3 Motivation

Local approaches do not consider entities that have temporal connections to the entities in the given pair at all, and thus contradictions within a document can occur. For instance, in the temporal relation classification task, the system may predict that A happens before B, that B happens before C, and that A happens after C, which are mutually contradictory.

Let's see an example sentence in Figure 1.4. The timeline of the temporal entities appearing in the sentence is shown on the left side of Figure 1.5.

If we look at the sentence, the words *one year* and the word *killed* are close to each other and the event *attend* and the time expression *one year*are also closed, so it is not hard to predict the relation type. But for the events *attend* and *kill*, they are distant and it is hard to predict their relationship by using only their local information, e.g. surrounding words, Part of Speech (POS) tags, or their tenses.

This is our motivation to put all of these temporal entities into a graph, as illustrated in Figure 1.6, and thus we can use the information extracted from the graph in the temporal

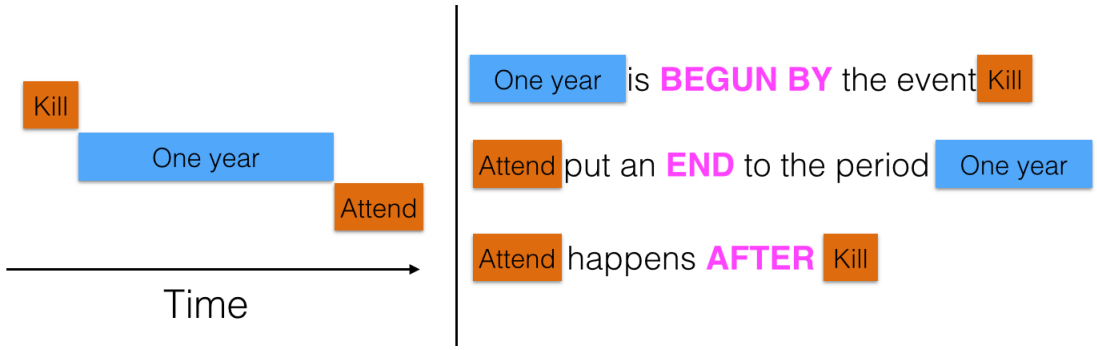


Figure 1.5: Temporal relation classification time line example (pairwise)

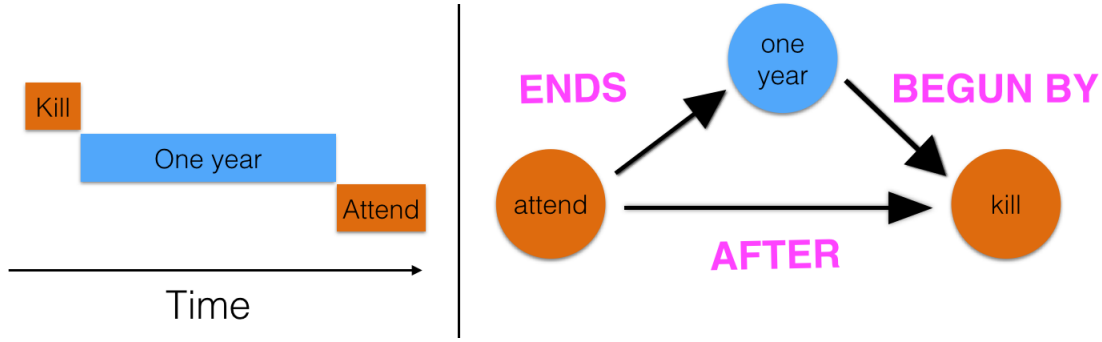


Figure 1.6: Temporal relation classification time line example (graph)

relation classification.

By considering the relations in a document as a graph consisting of nodes (entities) and edges (relations), we are able to extract more information such as *paths* between a pair of entities or *context* of a node, as illustrate in Figure 1.7. We call these kind of information *Non-Local Information*.

1.4 Non-local approaches

There exist many machine learning approaches that are capable of incorporating non-local information into relation extraction models, such as

- Structured perceptron (with global features)

  Structured perceptron is suitable for incorporating non-local features. However, high computation cost cannot be avoided.

- Conditional Random Field (CRF)

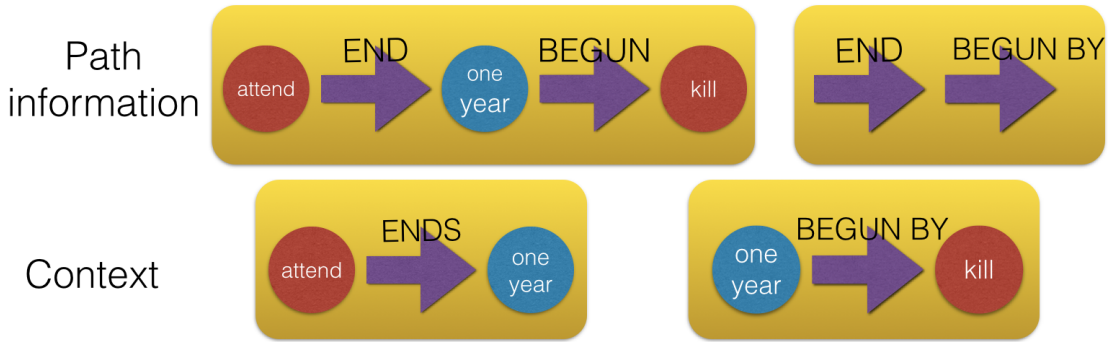


Figure 1.7: Non-local information

Normally, CRF is suitable for employing short distance features. Even we can increase the degree of the model but the computation cost will increase drastically.

- Markov Logic Network (MLN)

  MLN allows us to model the dependencies between entities but the number of features is quite limited. A very-long running time is expected if too many features are used.

- Reranking

  Reranking is suitable for incorporating non-local features. However, the best the reranking model is limited by the original model.

- Stacked learning

  Stacked learning is also suitable for incorporating non-local features. It employs multiple stages of learners (no form restriction) and the output of the current stage will be the input to the next stage. Each stage can use any kind to classifiers so we can choose the learning approach that is not computational expensive.

Since we want to apply a large number of features to our model, the stacked learning approach seems to be the most suitable choice for our requirement. So, in this study, in order to apply non-local information into the classification, we make use of graphs and apply a stacked learning method to classification task.

1.5 Contributions

The contributions of this work are as follows:

- Incorporate non-local information into the temporal relation classification task

  We built a stacked learning model and incorporated several types of non-local features into the model.

- Apply some inference methods to increase accuracy

  To alleviate the graph sparsity problem, we apply temporal relation inference and time-time connection to the graphs before extracting non-local features.

- Use probability values as real-valued features

  The probability values of the prediction in the first stage (of the stacked learning) are applied in temporal relation classification as real-valued features and used for prediction in the second stage.

- Evaluation and result analysis

  The temporal relation classification has been performed with extensive experiments to verify the proposed method. The results were analyzed in detail and compared with the state of the art.

Even though the idea can also be applied to other NLP relation extraction tasks, this work mainly focuses on the temporal relation classification and tackle the problem of contradictory predictions by using a stacked learning approach. The prediction for a temporal relation is made by considering the consistency of possible relations between nearby entities. By performing 10-fold cross validation on the Timebank corpus, we achieve an F1 score of 60.25% based on the graph-based evaluation, which is 0.90 percentage points higher than that of the local approach, and outperform other systems that have been presented so far.

II

Machine learning

The machine learning approaches that are used in our work are supervised machine learning which will be introduced in Section 2.1, and stacked learning which will be explained in Section 2.2. The application of stacked learning to NLP tasks is described in Section 2.3.

2.1 Supervised Machine Learning

As illustrated in Figure 2.1, supervised machine learning is a learning method that based on labeled training example. In the training stage, features and labels (correct answers) are extracted from the training data. Then, the predicting model is trained from these features and labels by some machine learning algorithms. The output from training stage is a set of prediction rules (model). In the predicting stage (testing stage), a new example without label is given. A set of features are extracted from the example and the model then predicts the output from those features.

Supervised machine learning is widely used in NLP classification tasks. Figure 2.2 shows the application of it in NLP-related classification. The training examples are text or sentences from documents. The features commonly used in NLP tasks are word tokens, part of speech (POS) tags, lemmas, parse tree, synonyms, etc., and labels are correct (gold) relations.

2.2 Stacked learning

Stacked learning method is proposed by*Wolpert* (1992). It is a machine learning method that contains multiple stages of learners. In this work, we will fix the number of stages to 2. The prediction results of the first stage will be the input feature to the second stage as illustrated in Figure 2.3.

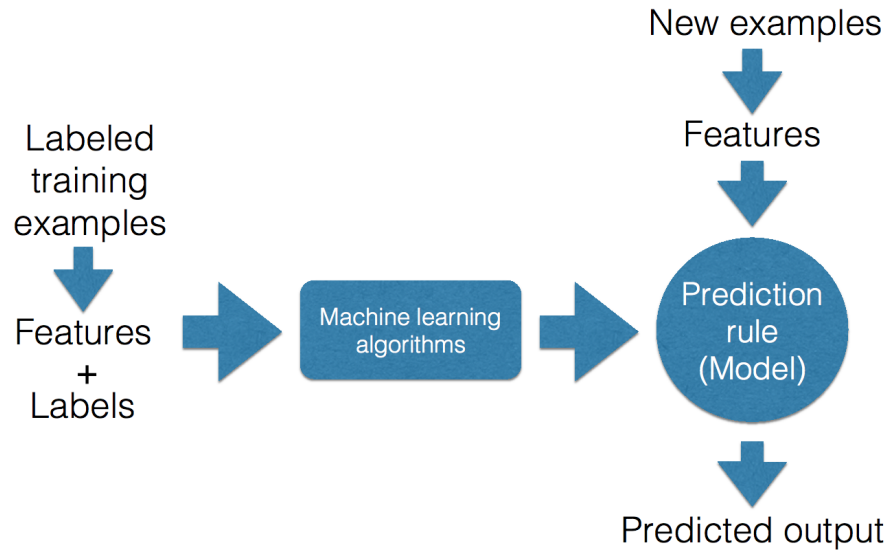


Figure 2.1: Supervised Machine Learning

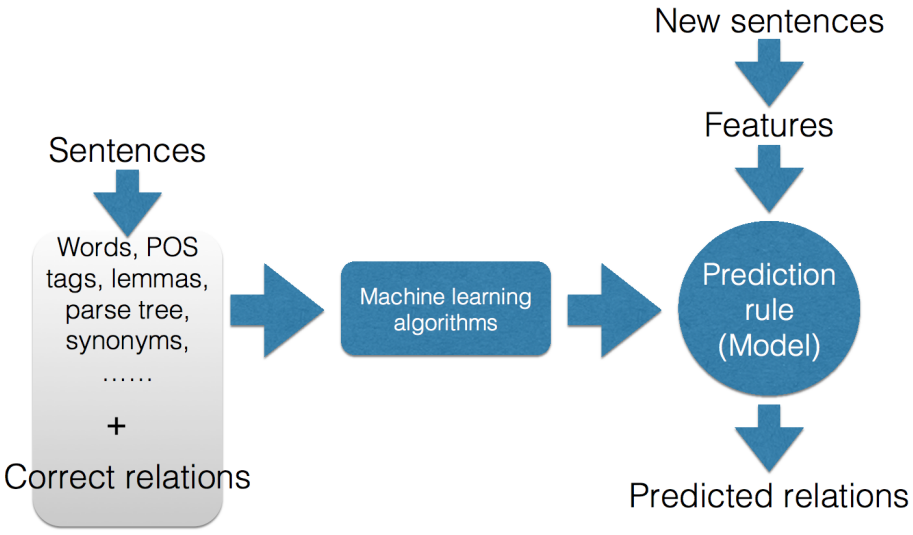


Figure 2.2: Supervised Machine Learning in NLP tasks

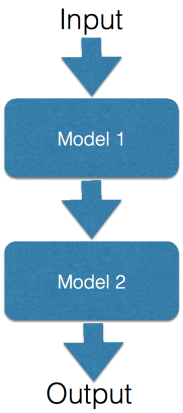


Figure 2.3: Stacked learning

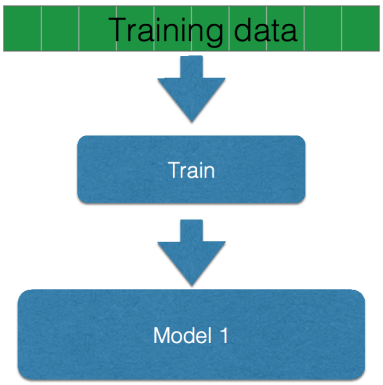


Figure 2.4: Training the first stage of the stacked learner

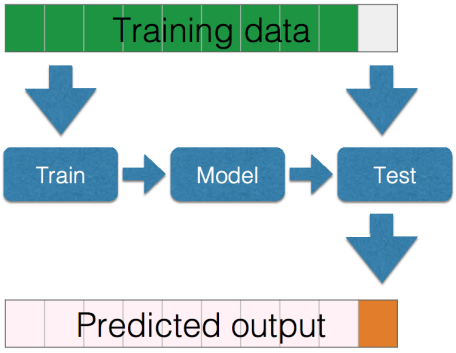


Figure 2.5: Ten-fold cross validation to create the training data for the second model (1)

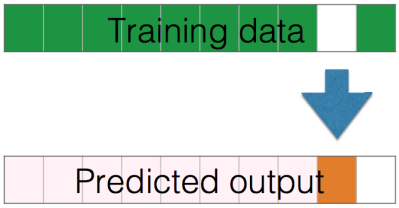


Figure 2.6: Ten-fold cross validation to create the training data for the second model (2)

The step-by-step training approach for the stacked learning is as follows.

First, we will use all training data to train the first model as illustrated in Figure 2.4. This model will be used in the test stage.

Second, in order to create the training data for the second model, 10-fold cross validation is performed. The prediction results from the 10-fold cross validation will be used as input to the second stage. This is because we do not want the second model to learn from the real (gold) labels of the training data. Figure 2.5, 2.6, and 2.7 illustrate this step.

Third, we use the predicted output from 10-fold cross validation to train the second model as represented in 2.8. After this step, the training for the stacked learner is already done.

In test stage, the first model obtained from the first step (Figure 2.4) is used to predict the output of the test data, as shown in Figure 2.9. Next, in the second stage illustrated in 2.10, the test data and the output from the first stage are used as input to the second model

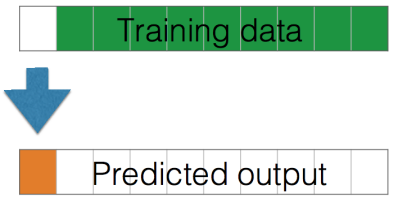


Figure 2.7: Ten-fold cross validation to create the training data for the second model (3)

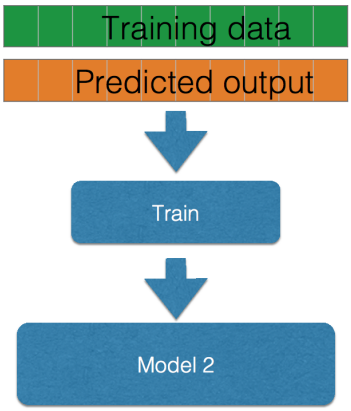


Figure 2.8: Training the second stage of the stacked learner

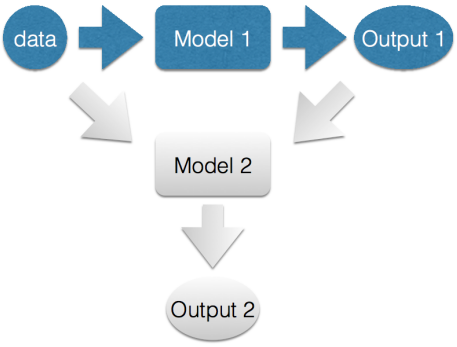


Figure 2.9: Test stage (first stage)

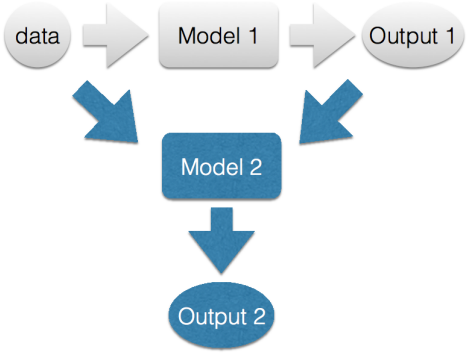


Figure 2.10: Test stage (second stage)

obtained in the third step (2.8). The second model again predicts the output which is the final result and may differ from the output of the first stage.

2.3 Stacked learning for NLP tasks

This section will show how to use the stacked learning approach in NLP tasks. First, let Figure 2.11 represent a general relation extraction task. *E* stands for *Entity* and *R* stands for *Relation*.

First, in Figure 2.12, the local features, such as words, lemmas, POS tags, and synonyms, are extracted from the entities and the first model is trained.

Second, with the same set of features extracted from the first step, 10-fold cross validation is performed as shown in Figure 2.13. After finishing the cross validation, all the relations have been predicted, as presented in 2.14.

Third, as illustrated in Figure 2.15, the relation graph is constructed from the output

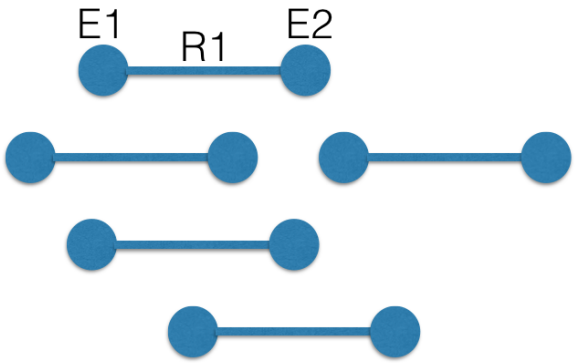


Figure 2.11: Relation extraction in NLP

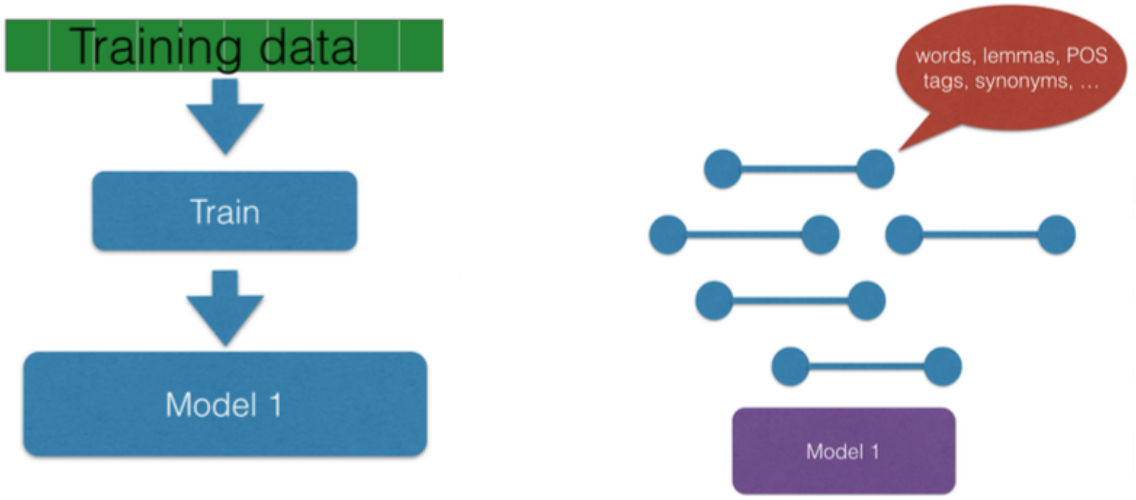


Figure 2.12: Training the first stage of the stacked learner for NLP tasks

of the first stage. Non-local features can be extracted from the relation graph. Then the second model is trained by using these non-local features as well as the same feature set extracted in the first stage, as we can see in Figure 2.16.

In the test stage in Figure 2.17, again the local features, e.g. words, lemmas, POS tags, and synonyms, are extracted from the test data. All the test relations are predicted using the first model, as shown in Figure 2.18.

After that, the relation graph is constructed as seen in Figure 2.19, and the non-local features are extracted. In the last step, the final prediction result can be obtained in Figure 2.20 by using the second model.

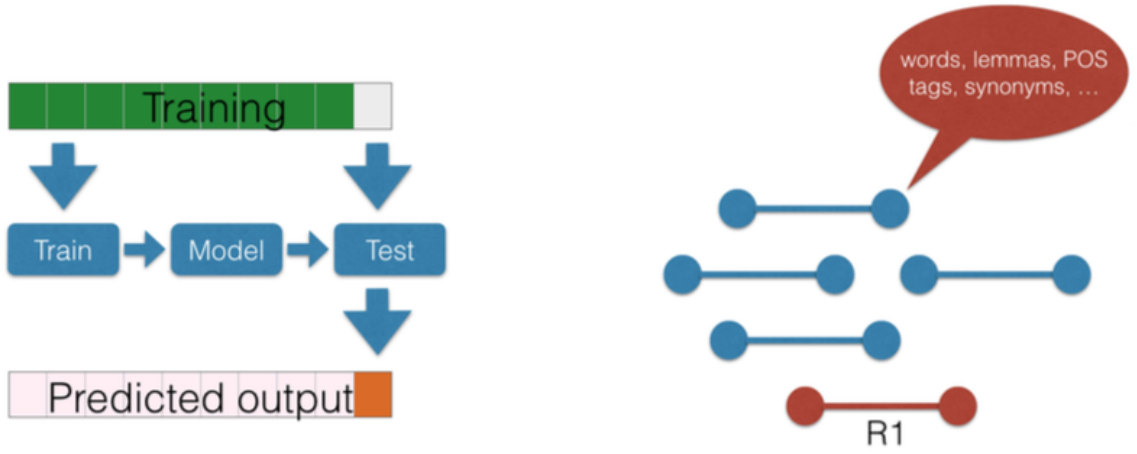


Figure 2.13: Ten-fold cross validation to create the training data for the second model in NLP tasks (1)

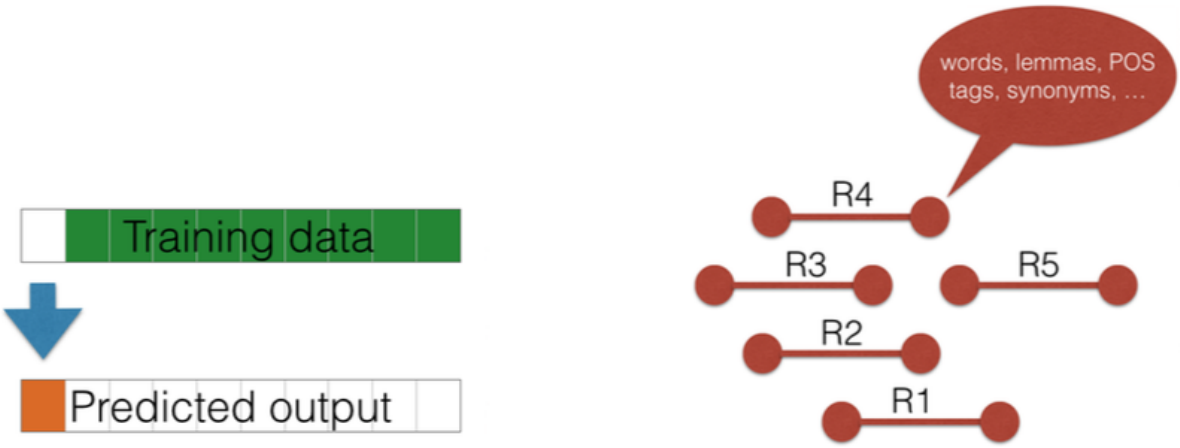


Figure 2.14: Ten-fold cross validation to create the training data for the second model in NLP tasks (2)

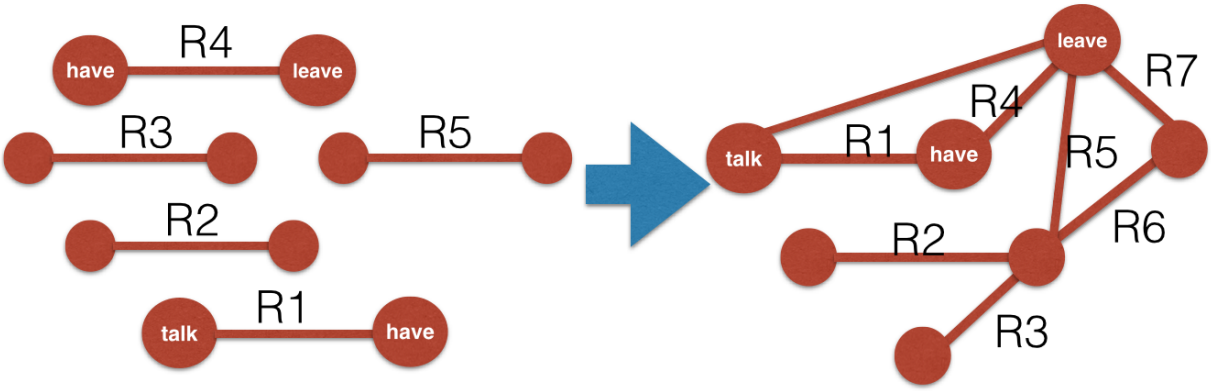


Figure 2.15: Constructing the relation graph from the prediction output of the first stage

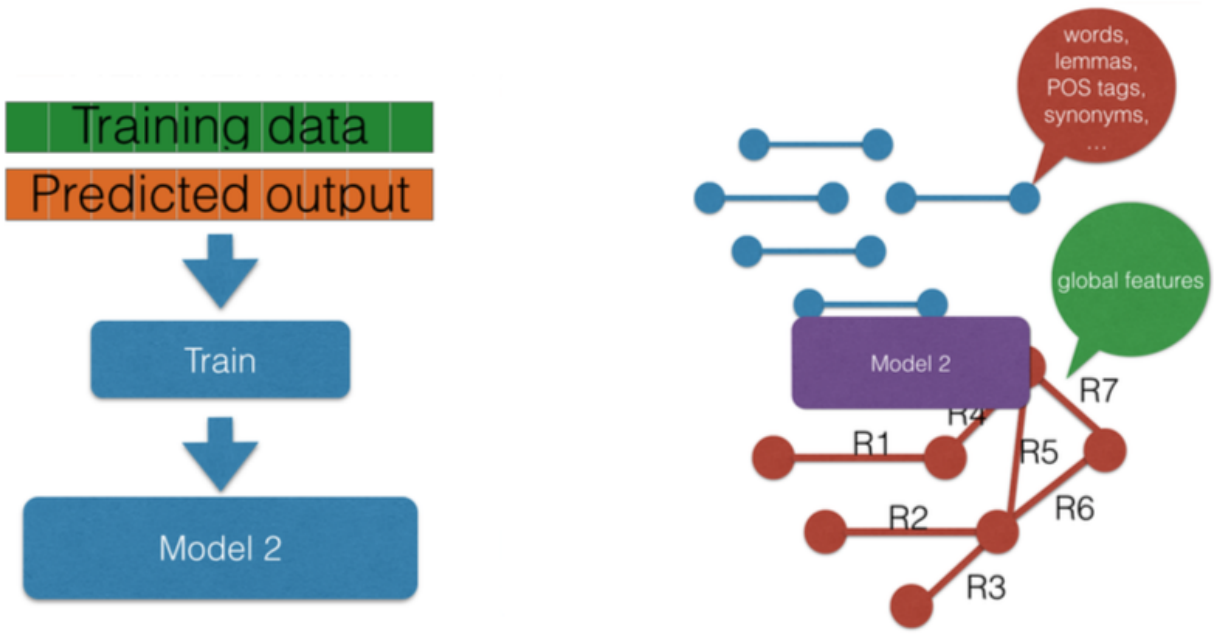


Figure 2.16: Training the second stage of the stacked learner for NLP tasks

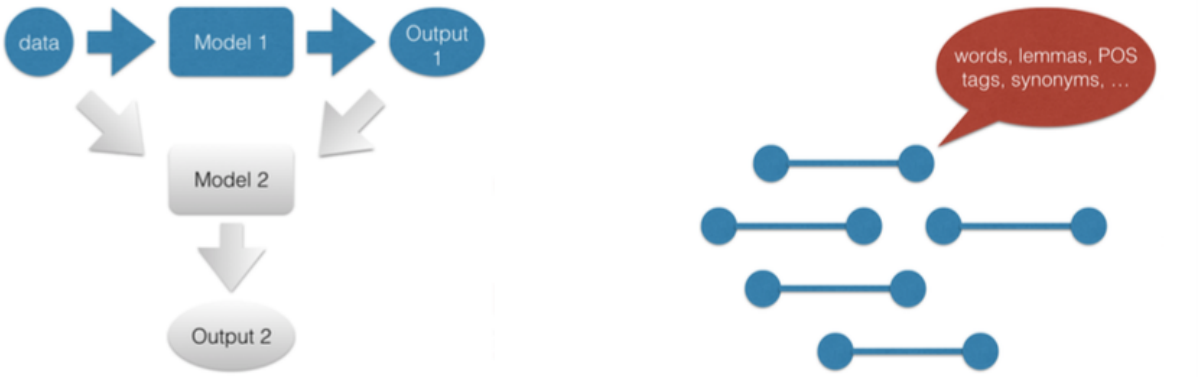


Figure 2.17: Test stage (first stage) (1)

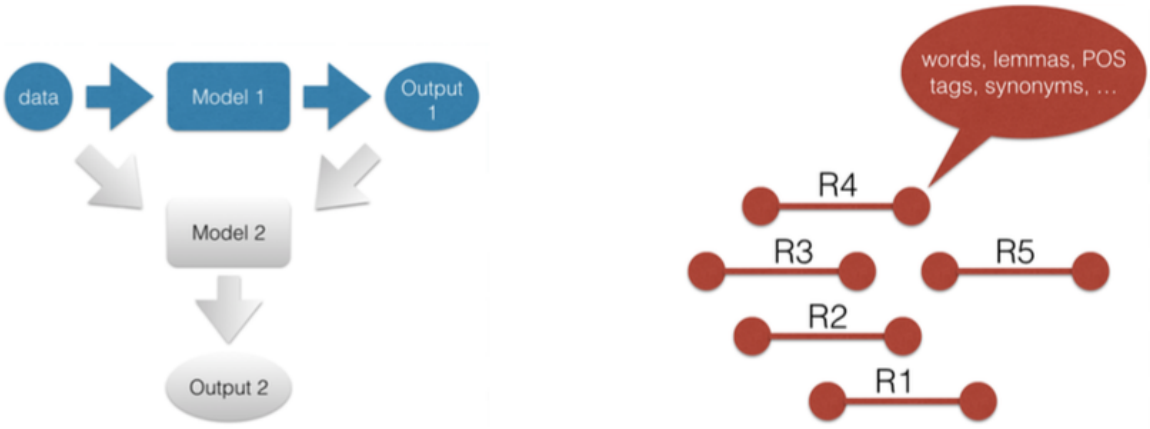


Figure 2.18: Test stage (first stage) (2)

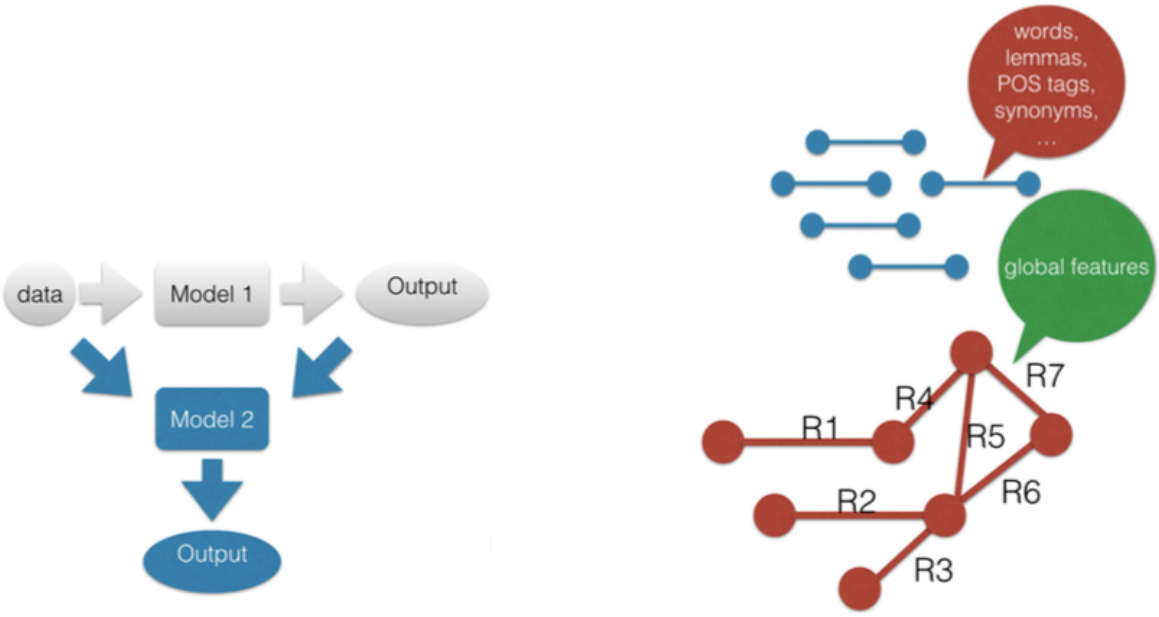


Figure 2.19: Test stage (second stage) (1)

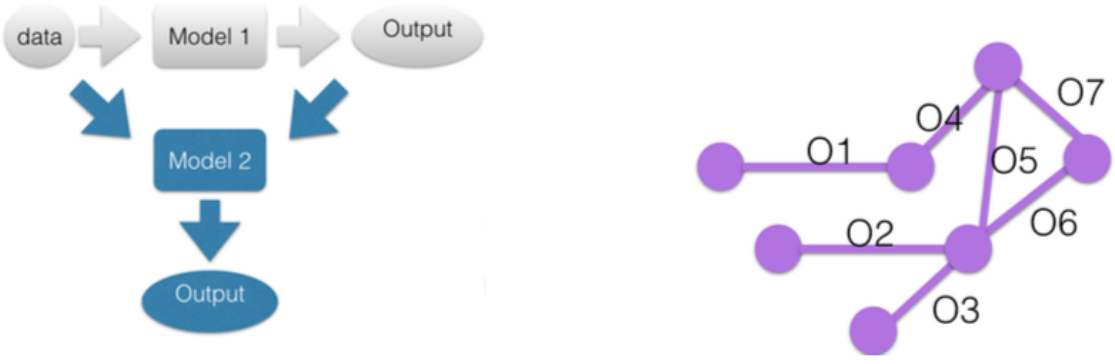


Figure 2.20: Test stage (second stage) (2)

III

Temporal relation classification

This chapter introduces the background about temporal relation classification and presents some previous studies related to our work.

3.1 Temporal Relation Classification

Temporal relations are an important piece of information for deep understanding of documents. Figure 3.1 shows an example of temporal relations in a news article. Temporal relation classification aims to classify temporal relationships between pairs of temporal entities into one of the relation types such as *BEFORE*, *AFTER*, *SIMULTANEOUS,* and *BEGINS.* Traditional approaches to temporal relation classification employ machine learning-based classifiers *Mani et al.* (2006) Being able to extract temporal relations between events and temporal expressions appearing in text is beneficial for various natural language processing (NLP) applications such as textual entailment *Bos and Markert* (2005), multi-document summarization *Bollegala et al.* (2006), and question answering *Ravichandran and Hovy*

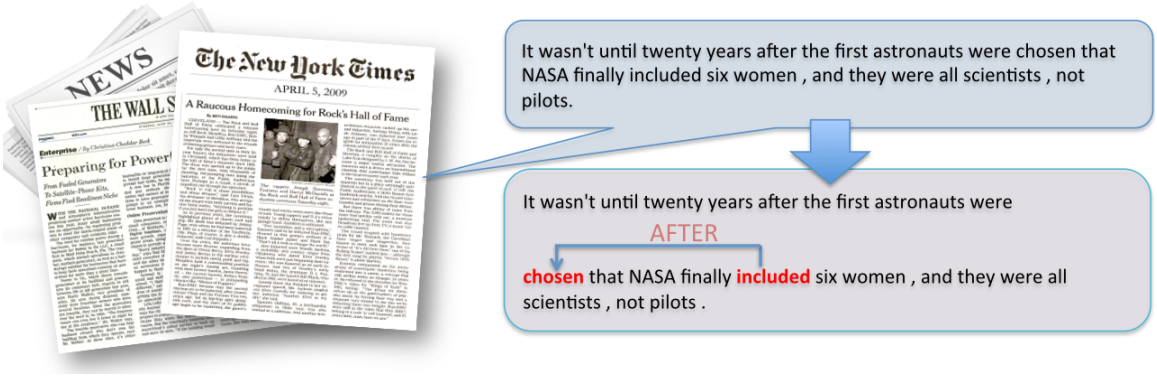


Figure 3.1: Temporal relations in a news article

(2002).

Temporal relation classification is one of the subtasks of TempEval challenge (*Verhagen et al.* (2010); *UzZaman et al.* (2012, 2013)). According to TempEval-3 which was held in 2013, a temporal annotation task can be separated into three subtasks:

- Temporal expression extraction (Task A)

  A task to extract temporal expressions appearing in a document, along with their attributes and normalized values. The state of the art uses the ruled-based approach (*Chang and Manning* (2012)).

- Event extraction (Task B)

  A task to extract events appearing in a document, along with their attributes. Many systems, such as the work of *UzZaman and Allen* (2010), used CRF for this task.

- Temporal link identification and relation classification (Task C)

  A task to identify whether two specific temporal entities has a temporal relation between them and classify the relation type into one if the 13 relations described in TimeML specification. There is another task called *Task C - relation only* which focuses only on relation classification by using the given pairs of temporal entities from the corpus.

3.1.1 Temporal expression

Temporal expressions are words that state

- A point of time, e.g. 5 minutes ago, yesterday
- A period or a range, e.g. one week or one year, 20 minutes)
- A frequency, e.g. everyday, every year

An example of temporal expressions in a sentences and their associated attributes are shown in Figure 3.2.

3.1.2 Event

Events are word tokens that represent states or occurence. They can be nouns, verb, numbers, or any kind of part of speech. An example of events in a sentences and their associated attributes are shown in Figure 3.3.

George Walker Bush (born July 6, 1946) is an American politician who served as the 43rd President of the United States from 2001 to 2009.

Event	Expression	Type	Normalized
Birth	July 6, 1946	DATE	1946-07-06
President	From 2001 to 2009	RANGE	2001/2009

Figure 3.2: Temporal expression example

He said that the flight would leave on Monday night.

Event	Class	Tense	Polarity	Part of speech	Modality
said	reporting	PAST	POS	VERB	-
leave	occurrence	PAST	POS	VERB	would

Figure 3.3: Event example

3.1.3 Temporal relation

Our work does not extract events and temporal expressions automatically but only focuses on the relation classification task (Task C and Task C-relation only). Figure 3.4 shows more example of temporal relations between temporal entities, including temporal expressions and events.

Following TempEval-3, we consider the following four kinds of temporal relations:

- A relation between an event and the *Document Creation Time (DCT)*,
- A relation between two events in the same sentence,
- A relation between an event and a temporal expression in the same sentence, and
- A relation between two events in consecutive sentences.

Figure 3.6 shows how temporal relations in the document in Figure 3.5 looks like. The document, events, temporal expressions, and the temporal relations are annotated and visualized by a visualization tool, BRAT invented by *Stenetorp et al.* (2012). Figure 3.7 shows the connection in the form of a graph consisting of nodes (temporal entities) and edges (temporal relations).

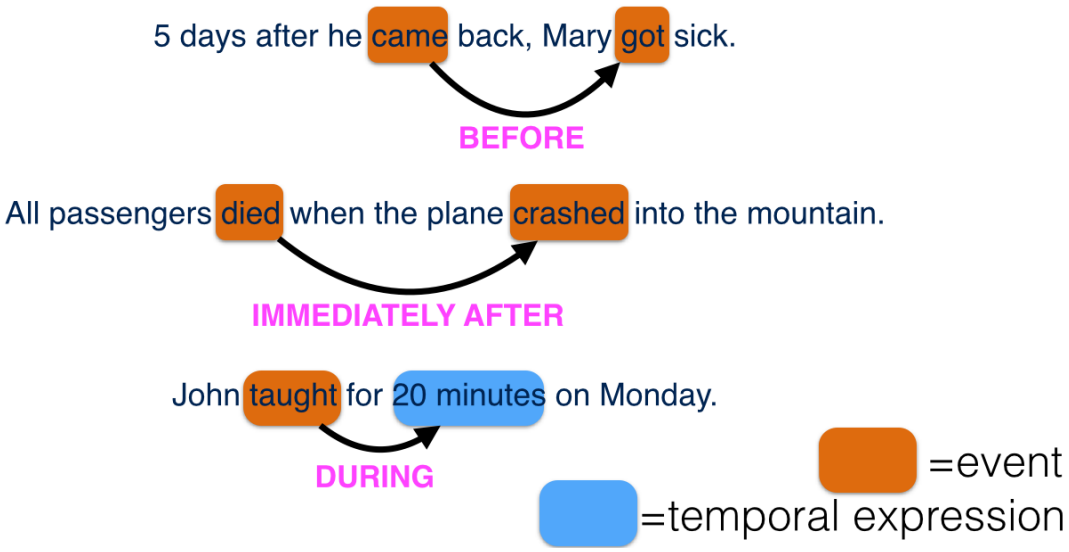


Figure 3.4: Temporal relation examples

```
Finally today , we learned that the space agency has finally taken a giant leap forward .
Air Force Lieutenant Colonel Eileen Collins will be named commander of the Space Shuttle Columbia for a mission in December .
Colonel Collins has been the co-pilot before , but this time she 's the boss .
Here 's ABC 's Ned Potter .
Even two hundred miles up in space , there has been a glass ceiling .
It was n't until twenty years after the first astronauts were chosen that NASA finally included six women , and they were all
scientists , not pilots .
No woman has actually been in charge of a mission until now .
Just the fact that we 're doing the job that we 're doing makes us role models .
That was Eileen Collins , after she flew as the first ever co-pilot .
Being commander is different .
It means supervising the rest of the crew in training and leading them in flight .
It is , in short , the kind of management job many American women say they 've had to fight for .
In space , some say female pilots were held up until now by the lack of piloting opportunities for them in the military .
Once Colonel Collins was picked as a NASA astronaut , she followed a normal progression within NASA .
Nobody hurried her up .
No one held her back .
Many NASA watchers say female astronauts have become part of the agency 's routine .
But they still have catching up to do two hundred and thirty four Americans have flown in space , only twenty six of them
women .
Ned Potter , ABC News .
```

Figure 3.5: A news article taken from the Timebank corpus (ABC19980304.1830.1636.tml)

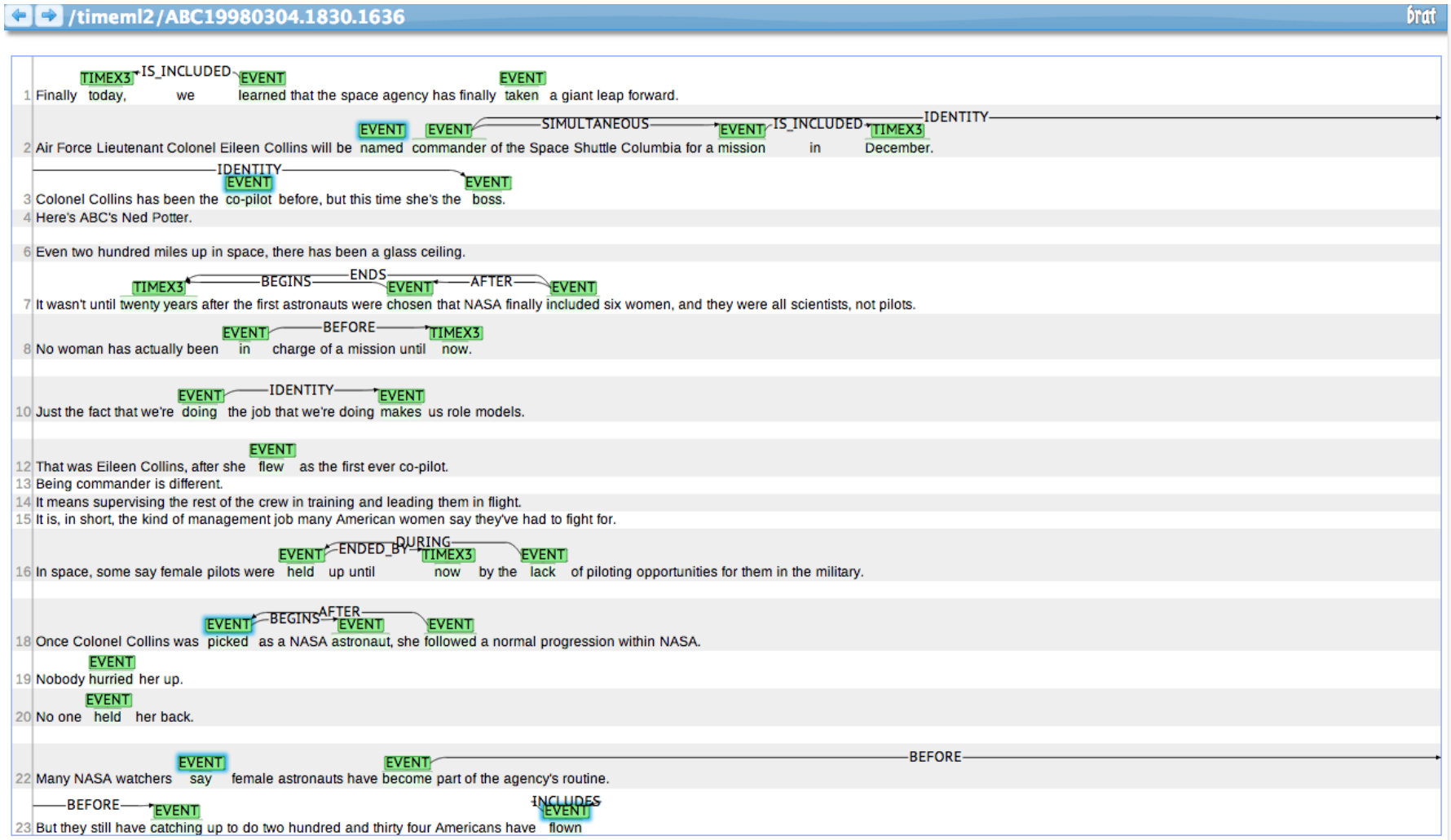


Figure 3.6: Temporal relation in a document

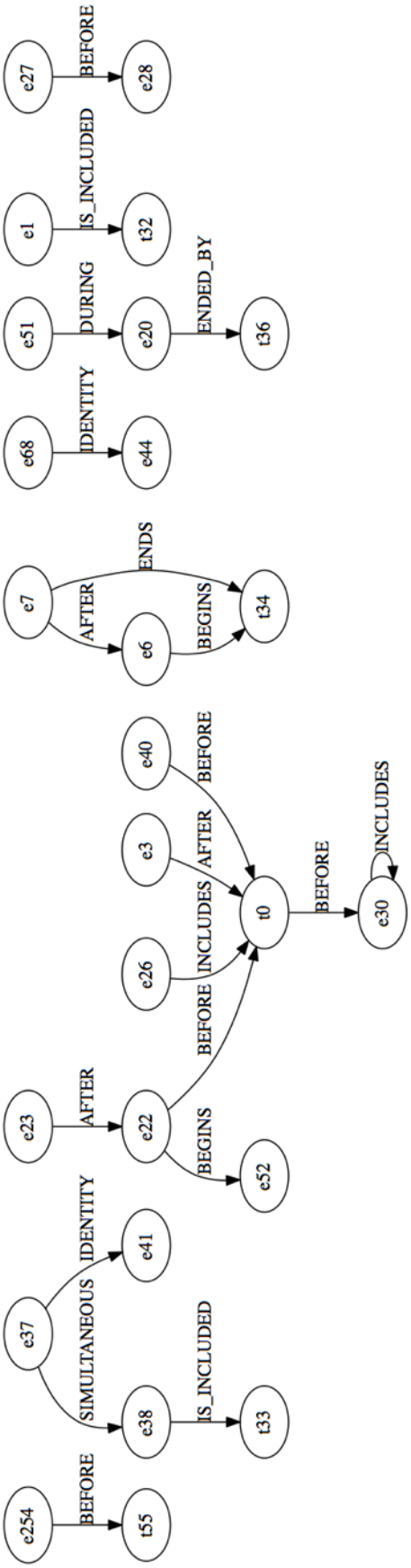


Figure 3.7: Temporal relation in a document (graph)

3.1.4 The Timebank corpus

The Timebank corpus, introduced by *Pustejovsky et al.* (2003), is a human-annotated corpus commonly used in training and evaluating a temporal relation classifier. It is annotated following the TimeML specification to indicate events, temporal expressions, and temporal relations.

It also provides five attributes, namely, *class, tense, aspect, modality,* and *polarity*, associated with each event *(EVENT)*, and four attributes, namely, *type, value, functionInDocument,* and *temporalFunction*, associated with each temporal expression *(TIMEX3)*. A TIMEX tag can be used to annotate a date, time, duration, and a set of dates and times. An example of the annotated event and temporal expression is shown below. The sentence is brought from wsj_0292.tml in the Timebank corpus.

> *In <TIMEX3 tid="t88" type="DURATION" value="P9M" temporalFunction="true" functionInDocument="NONE" endPoint="t0">the first nine months</TIMEX3>, profit <EVENT eid="e30" class="OCCURRENCE">rose</EVENT> 10% to $313.2 million, or $3.89 a share, from $283.9 million, or $3.53 a share.*
>
> *<MAKEINSTANCE eventID="e30" eiid="ei349" tense="PAST" aspect="NONE" polarity="POS" pos="VERB" />*

In addition to an *EVENT* tag which includes a *class* attribute, an event is also annotated with one or more *MAKEINSTANCE* tags that include information about a particular instance of the event. The information includes *tense, aspect, modality,* and *polarity*. In the above example, there is no modal word in the sentence, so the attribute *modality* does not appear.

A pair of temporal entities, including events and temporal expressions, that is annotated as a temporal relation is called a TLINK. Temporal relation classification is the task of classifying TLINKs into temporal relation types. We use the complete set of the TimeML relations, which has 14 types of temporal relations: *BEFORE, AFTER, IMMEDIATELY BEFORE, IMMEDIATELY AFTER, INCLUDES, IS INCLUDED, DURING, DURING INVERSE, SIMULTANEOUS, IDENTITY, BEGINS, BEGUN BY, END,* and *ENDED BY.* However, in TempEval-3, *SIMULTANEOUS* and *IDENTITY* are regarded as the same relation type, so we replace all *IDENTITY* relations with *SIMULTANEOUS*.

Given the example mentioned above, the temporal relation is annotated as below.

> *<TLINK lid="l23" relType="DURING" eventInstanceID="ei349" relatedToTime="t88" />*

Relation	Example
before	5 days after he came back Mary got sick.
after	The group began to fall apart after the defection of its top leader.
ibefore	All passengers died when the plane crashed into the mountain.
iafter	An off duty policeman is killed when a bomb explodes.
includes	He had been working with the company when he was abducted.
is included	He said the stock won't cover the cost of building the resort, and added it will need to seek additional financing.
begins	Interviews with three major fund groups confirm the trend.
begun by	He said he will begin withdrawing the troops from the territory.
ends	It expects to complete the transaction by the year-end.
ended by	They will stay there until the crisis ends.
during	John taught for 20 minutes on Monday.
during inv	John taught for 20 minutes on Monday.
simultaneous	When he spent 70 million dollars for this house, he thought it was a great deal.
identity	John drove to Boston. During his drive he ate a donut.

Figure 3.8: Relation types

From the annotated relation above, the event **rose (e30)** happens *DURING* the temporal expression **the first nine months (t88)**.

Figure 3.8 shows all relation types along with their example sentences.

3.2 Related work

This section describes existing approaches to temporal relation classification along with their advantages and disadvantages.

Global approaches for tackling the aforementioned problem have been proposed previously (*Chambers and Jurafsky*, 2008; *Yoshikawa et al.*, 2009; *Denis and Muller*, 2011; *Do et al.*, 2012). Chambers and Jurafsky used Integer Linear Programming (ILP) to maximize the confidence scores of the output of local classifiers in order to solve the contradictory prediction.

Denis and Muller also used ILP but they enforced temporal relation coherence only on particular sets of events rather than on the entire documents. However, both of the studies focused only on the temporal relations between events and used reduced sets of the

temporal relations, i.e., Chambers and Jurafsky used *BEFORE*, *AFTER*, and *VAGUE.*, while Denis and Muller used *BEFORE*, *AFTER*, *OVERLAP*, and *NO RELATION.*

Do et al. employed a full set of temporal relations to construct a globally coherent timeline for an article using ILP, leveraged event coreference to support timeline construction, and associated each event with a precise time interval.

Yoshikawa et al. proposed a Markov Logic model to jointly predict the temporal relations between events and time expressions. They also used a reduced set of the relation types, i.e., *BEFORE*, *OVERLAP*, *AFTER*, *BEFORE-OR-OVERLAP*, *OVERLAP-OR-AFTER*, and *VAGUE*.

IV

Local approach to temporal relation classification

This chapter presents a local approach to temporal relation classification in Section 4.1 as well as the baseline and deep features in Section 4.1 and 4.3. The relation identification methods are also introduced in Section 4.4 and 4.5.

4.1 Local model

In the local model, temporal relation classification is done by considering only a given pair of temporal entities at a time as illustrated in Figure 4.1. We use a supervised machine learning approach and employ the basic feature set that can be easily extracted from the document's text and the set of features proposed in *Laokulrat et al.* (2013b), which utilizes deep syntactic information. The baseline features at different linguistic levels are listed in Section 4.2. For every TLINK, the baseline features include event attributes, time attributes, morphosyntactic information, lexical semantic information, and deep syntactic information. In addition, for Event-Event TLINKs, the matching of event attributes is also used as local features.

Two classifiers are used; one for Event-Event TLINKs (E-E), and the other one for Event-Time TLINKs (E-T).

4.2 Baseline features

The baseline features we employed are:

- Event and timex attributes

  All attributes associated with events (class, tense, aspect, modality, and polarity) and temporal expressions (type, value, functionInDocument, and temporalFunction) are

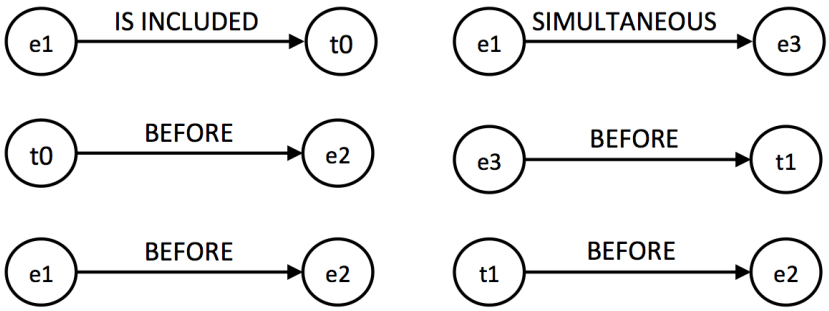


Figure 4.1: Local pairwise classification

used. For event-event TLINKs, we also use tense/class/aspect match, tense/class/aspect bigrams as features (*Chambers et al.* (2007)).

- Morphosyntactic information

  Words, part of speech tags, lemmas within a window before/after event words are extracted using Stanford coreNLP invented by *Manning et al.* (2014).

- Lexical semantic information

  Synonyms of event word tokens from WordNet lexical database invented by *Fellbaum* (2010) are used as features.

- Event-Event information

  For event-event TLINKs, we use *same_sentence* feature to differentiate pairs of events in the same sentence from pairs of events from different sentences (*Chambers et al.* (2007)).

We directly read the attributes as tagged in the corpus, different from some of the related work that they automatically determine the attributes by Support Vector Machine (SVM).

4.3 Features extracted from a deep syntactic parser

In the case that temporal entities of a particular TLINK are in the same sentence, we extract two new types of sentence-level semantic information from a deep syntactic parser. We use the Enju parser which was invented by *Miyao and Tsujii* (2008). It analyzes syntactic/semantic structures of sentences and provides phrase structures and predicate-argument structures. The features we extract from the deep parser are

- Paths between event words in the phrase structure tree, and up(↑)/down(↓) lengths of paths.

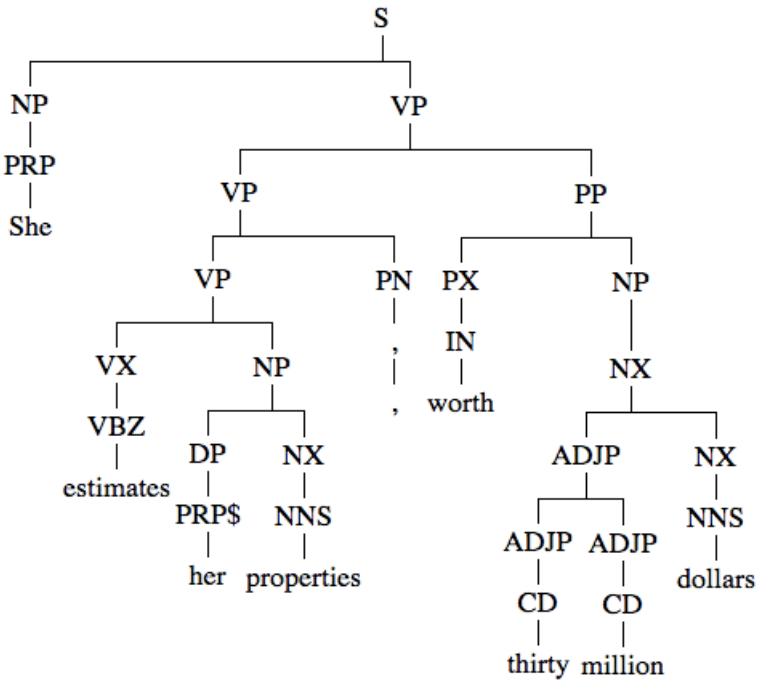


Figure 4.2: Phrase structure tree

We use 3-grams of paths as features instead of full paths since these are too sparse. An example is shown in Figure 4.2. In this case, the path between the event words, *estimates* and *worth*, is *VBZ↑, VX↑, VP↑, VP↑, VP, PP↓, PX↓, IN↓*. The 3-grams of the path are, therefore, {*VBZ↑-VX↑-VP↑, VX↑-VP↑-VP↑, VP↑-VP↑-VP, VP↑-VP-PP↓, VP-PP↓-PX↓, PP↓-PX-↓-IN↓*}. The up/down path lengths are 4 *(VBZ↑, VX↑, VP↑, VP↑)* and 3 *(PP↓, PX↓, IN↓)* respectively.

- Paths between event words in predicate-argument structure, and their subgraphs.

  For the previous example, we can express the relations in predicate-argument structure representation as

  - verb_arg12: estimate (she, properties)
  - prep_arg12: worth (estimate, dollars)

  In this case, the path between the event words, *estimates* and *worth*, is ←*prep_arg12:arg1*. That is, the type of the predicate *worth* is *prep_arg12* and it has *estimate* as the first argument (arg1). The path from *estimate* to *worth* is in reverse direction (←).

  The next example sentence, *John saw mary before the meeting*, gives an idea of a

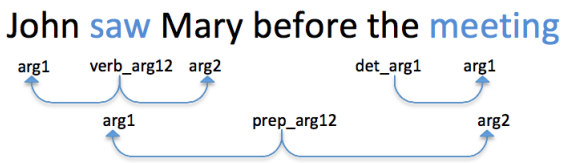


Figure 4.3: Predicate argument structure

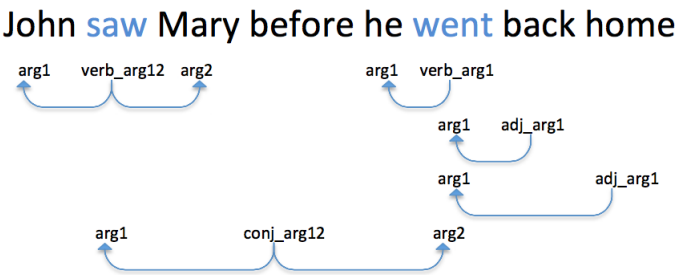


Figure 4.4: Predicate argument structure (2)

more complex predicate-argument structure as shown in Figure 4.3. The path between the event words, *saw* and *meeting* is *←prep_arg12:arg1, prep_arg12:arg2*.

We use (v, e, v) and (e, v, e) tuples of the edges and vertices on the path as features. For example, in Figure 4.3, the (v,e,v) tuples are *(see, ←prep_arg12:arg1, before)* and *(before, prep_arg12:arg2, meeting)*. In the same way, the (e,v,e) tuple is *(←prep_arg12:arg1, before, prep_arg12:arg2)*. The subgraphs of (v, e, v) and (e, v, e) tuples are also used, including *(see, ←prep_arg12:arg1, *), (*, ←prep_arg12:arg1, before), (*, ←prep_arg12:arg1, *), (*, prep_arg12:arg2, meeting), (before, prep_arg12:arg2, *), (*, prep_arg12:arg2, *), (*, before, prep_arg12:arg2), (←prep_arg12:arg1, before, *), (*, before, *)*.

From the above example, the features from predicate argument structure can properly capture the preposition *before*. It can also capture a preposition from a compound sentence such as *John saw Mary before he went back home*. As shown in Figure 4.4, the path between the event words *saw* and *went* are *(←conj_arg12:arg1, conj_arg12:arg2)* and the (v, e, v) and (e, v, e) tuples are *(saw, ←conj_arg12:arg1, before)*, *(before, conj_arg12:arg2, went)*, and *(←prep_arg12:arg1, before, prep_arg12:arg2)*.

4.4 TLINK identification

A pair of temporal entities that have a temporal relation is called a TLINK. The system first determines which pairs of temporal entities are linked by using a ruled-based approach as a baseline approach.

All the TempEval-3's possible pairs of temporal entities are extracted by a set of simple rules; pairs of temporal entities that satisfy one of the following rules are considered as TLINKs.

- Event and document creation time
- Events in the same sentence
- Event and temporal expression in the same sentence
- Events in consecutive sentences

4.5 Hybrid approach

The rule-based approach described in Section 4.4 produces many unreasonable and excessive links. We thus use a machine learning approach to filter out those unreasonable links by training the model in Section 4.1 with an additional relation type, *UNKNOWN*, for links that satisfy the rules in Section 4.4 but do not appear in the training data.

In this way, for Task C, we first extract all the links that satisfy the rules and classify the relation types of those links. After classifying temporal relations, we remove the links that are classified as *UNKNOWN*. The system overview can be found in Figure 4.5.

4.6 Evaluation

Two L2-regularized logistic regression classifiers, LIBLINEAR invented by *Fan et al.* (2008), are used; one for event-event TLINKs, and another one for event-time TLINKs. In addition to baseline features at different linguistic levels, features extracted by a deep syntactic parser are used.

The scores are calculated by the graph-based evaluation metric proposed by *UzZaman and Allen* (2011). We trained the models with TimeBank and AQUAINT corpora. We also trained our models on the training set with inverse relations. The performance analysis is based on 10-fold cross validation on the development data.

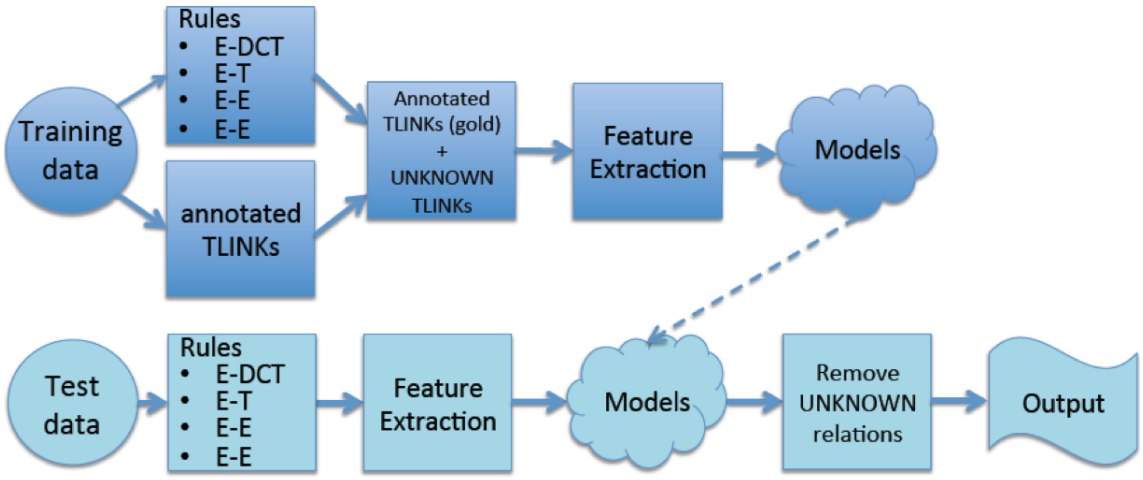


Figure 4.5: TLINK identification and classification system overview

Features	F1	P	R
bas. (rule)	22.51	14.32	52.58
bas. + ph. + pas. (rule)	22.61	14.30	54.01
bas. + ph. + pas. (hyb.)	33.52	36.23	31.19
bas. + ph. + pas. (hyb. + inv.)	39.53	37.56	41.70

Table 4.1: Result of Task C. (rule: rule-based approach, hyb.: hybrid approach, bas.: baseline features, ph.:phrase structure tree features, pas.:predicate-argument structure features, and inv.: Inverse relations are used for training.)

4.6.1 Task C

In Task C, a system has to identify appropriate temporal links and to classify each link into one temporal relation type. For Task C evaluation, we compare the results of the models trained with and without the features from the deep parser. The results are shown in Table 4.1. The rule-based approach gives a very low precision.

4.6.2 Task C-relation-only

Task C-relation-only provides a system with all the appropriate temporal links and only needs the system to classify the relation types. Since our goal is to exploit the features from the deep parser, in Task C-relation-only, we measured the contribution of those features to temporal relation classification in Table 4.2.

Features	F1	P	R
bas.	64.42	64.59	64.25
bas. + ph.	65.24	65.42	65.06
bas. + pas.	66.40	66.55	66.25
bas. + ph. + pas.	66.39	66.55	66.23
bas. + ph. + pas. (inv.)	65.30	65.39	65.20

Table 4.2: Result of Task C-relation-only. (bas.: baseline features, ph.:phrase structure tree features, pas.:predicate-argument structure features, and inv.: Inverse relations are used for training.)

4.6.3 Results on test data

We named our system *UTTime*, which has several combination of configurations and weights.

Table 4.3 and 4.4 show the results on the test data, which were manually annotated and provided by the TempEval-3 organizer. We also show the scores of the other systems in the tables and illustrate the results in Figure and . For the evaluation on the test data, we used the models trained with baseline features, phrase structure tree features, and predicate-argument structure features.

UTTime-5 ranked 2^{nd} best in Task C. Interestingly, training the models with inverse relations improved the system only when using the hybrid approach. This means that the inverse relations did not improve the temporal classification but helped the system filter out unreasonable links (UNKNOWN) in the hybrid approach. As expected, the ruled-based approach got a very high recall score at the expense of precision. UTTime-1, although it achieved the F1 score of only 24.65, got the highest recall among all the systems.

The system description for NavyTime and JU-CSE can be found in *Chambers* (2013) and *Kolya et al.* (2013).

For Task C-relation-only, we achieved the highest F1 score, precision, and recall. UTTime-2 basically had the same models as that of UTTime-1, but we put different weights for each relation type. The results show that using the weights did not improve the score in graph-based evaluation.

Approach	F1	P	R
rule (UTTime-1)	24.65	15.18	65.64
rule + inv (UTTime-3)	24.28	15.1	61.99
hyb. (UTTime-4)	28.81	37.41	23.43
hyb. + inv. (UTTime-5)	34.9	35.94	33.92
cleartk	**36.26**	**37.32**	**35.25**
NavyTime	31.06	35.48	27.62
JU-CSE	26.41	21.04	35.47
KUL-KULTaskC	24.83	23.35	26.52

Table 4.3: Result of Tack C on test data. (rule: rule-based approach, hyb.: hybrid approach, and inv.: Inverse relations are used for training.)

Approach	F1	P	R
bas. + ph. + pas. (UTTime-1)	**56.45**	**55.58**	**57.35**
bas. + ph. + pas. (UTTime-2)	54.26	53.2	55.36
bas. + ph. + pas. (inv.) (UTTime-3)	54.7	53.85	55.58
NavyTime	46.83	46.59	47.07
JU-CSE	34.77	35.07	34.48

Table 4.4: Result of Task C-relation-only on test data. (bas.: baseline features, ph.:phrase structure tree features, pas.:predicate-argument structure features, and inv.: Inverse relations are used for training.)

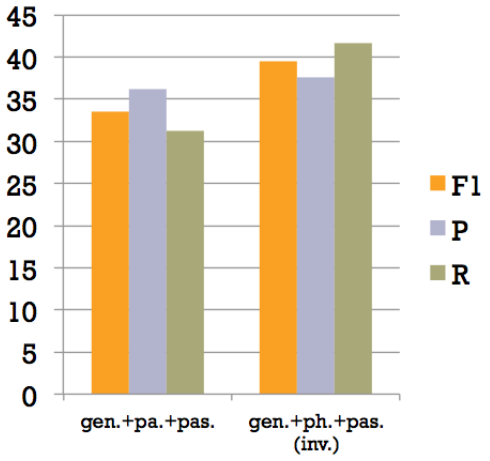


Figure 4.6: Results of Task C (TLINK identification and classification)

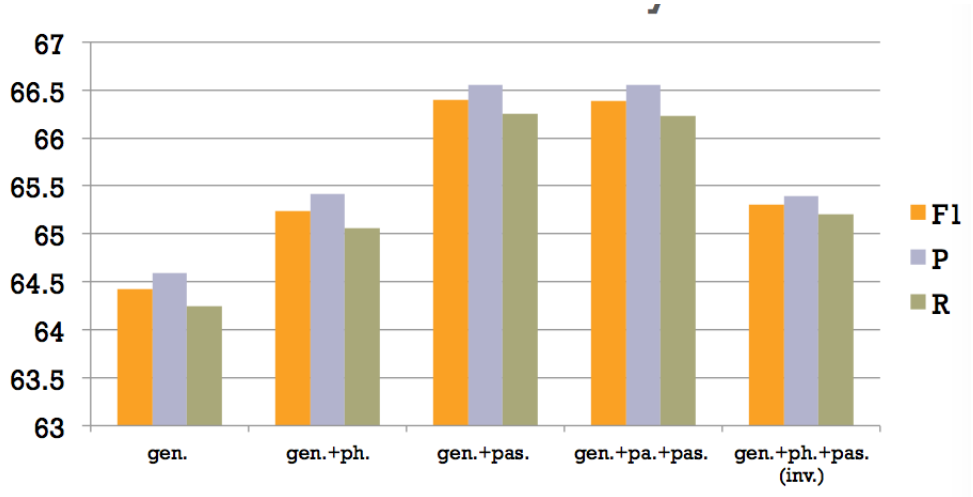


Figure 4.7: Results of Task C - relation only (TLINK classification)

V

Non-local approach to temporal relation classification

This chapter explains our stacked learning approach for temporal relation classification. The local model (first stage of the stacked learning) was introduced in Section 4.1 and the non-local model (second stage) will be described in Subsection 5.1.1.

5.1 Stacked model for temporal relation classification

Rather than using only local information on two entities in a TLINK as summarized in Table 5.1, we exploit more global information which can be extracted from a document's timegraph, a graph that represents temporal entities and their relations. Our motivation is that temporal relations of nearby TLINKs in a timegraph provide useful information for predicting the relation type of a given TLINK. For instance, consider the following sentence and the temporal connectivity shown in Figure 5.2, which is constructed from the temporal entities and temporal relations appearing in the sentence.

The first stage, as shown in Figure 4.1, uses the local classifiers and predicts the relation types of all TLINKs.

> *About 500 people **attended (e1)** a Sunday night memorial for the Buffalo-area physician who performed abortions, **one year (t1)** after he was **killed (e2)** by a sniper's bullet.*

It can be seen that the relation between ***e1*** and ***t1*** and the relation between ***t1*** and ***e2*** are useful for predicting the relation between ***e1*** and ***e2***. That is to say, if we know the fact that ***e1*** put an end to the period ***t1***, and the period ***t1*** started from the occurrence of ***e2***, we can infer that ***e1*** must happened after ***e2***.

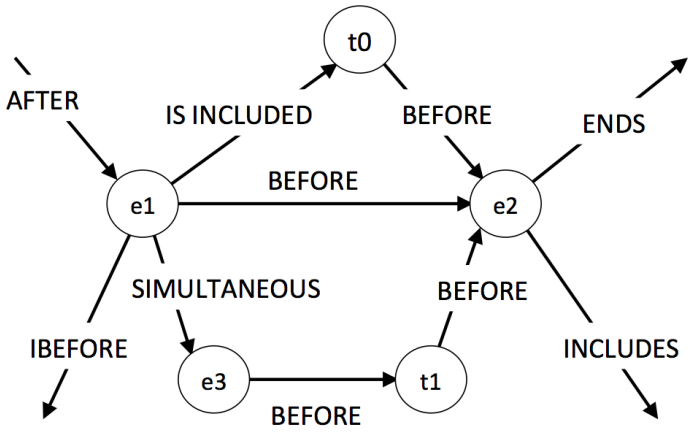


Figure 5.1: Timegraph

Feature	Description
Event attributes	All attributes associated with events, i.e., *Class, Tense, Aspect, Modality,* and *Polarity*. The explanation of each attribute can be found in *Pustejovsky et al.* (2005).
Timex attributes	All attributes associated with temporal expressions, i.e., *Type, Value, FunctionInDocument,* and *TemporalFunction*. The explanation of each attribute can also be found in *Pustejovsky et al.* (2005).
Morphosyntactic information	*Words, POS tags,* and *lemmas* within a window before/after event words. We extracted those information by using Stanford coreNLP *Manning et al.* (2014).
Lexical semantic information	*Synonyms* of event word tokens and temporal expressions extracted from WordNet lexical database *Fellbaum* (2010).
Event-Event information	*Class match, Tense match, Aspect match, Class bigram, Tense bigram, Aspect bigram,* and *Same sentence flag* (True if both temporal entities are in the same sentence). The details are described in *Chambers et al.* (2007). Note that these features are only applied in the E-E classifier.
Deep syntactic information	Deep syntactic information extracted from Enju Parser *Miyao and Tsujii* (2008). The features are paths between pairs of temporal entities in phrase structure and in predicate-argument structure. The details are described in *Laokulrat et al.* (2013b). Note that these features can be extracted only when temporal entities are in the same sentences.

Table 5.1: Local (baseline + deep) features

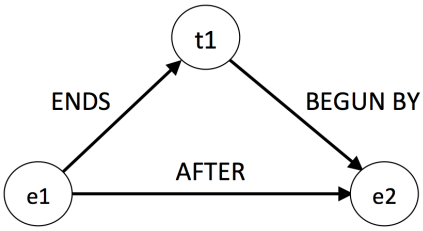


Figure 5.2: Temporal relations. Path length ≤ 2

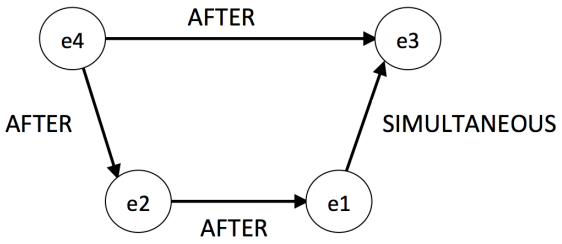


Figure 5.3: Temporal relations. Path length ≤ 3

Figure 5.3 shows another example of a group of temporal relations that has a longer path. In this example, the relation between ***e4*** and ***e3*** can be inferred from the nearby relations, i.e., (1) ***e4*** *AFTER* ***e2*** and ***e2*** *AFTER* ***e1*** imply ***e4*** *AFTER* ***e1***, (2) ***e4*** *AFTER* ***e1*** and ***e1*** *SIMULTANEOUS* ***e3*** imply ***e4*** *AFTER* ***e3***.

There are many machine learning approaches that can exploit global features such as stacked learning, Markov Logic Networks, and structured perceptron. However, each method has its advantages and disadvantages. A Markov Logic Network can model dependencies between temporal entities easily but it has a limitation in employing high-dimensional features, e.g. bag of words and n-grams, which are considered to be very powerful features for temporal relation classification. In a structured perceptron, computation time increases exponentially with the number of possible labels, unless the structure is simple enough for one to employ an efficient searching algorithm, e.g. Viterbi algorithm, to find the best solution. A stacked learning method enables a learner to be aware of labels of nearby entities by employing more than one stages of learning. In each stage, we can use any kind of efficient learners that can handle high-dimensional features, such as support vector machines (SVMs) and logistic regression, so it is able to employ rich linguistic features and, at the same time, exploit global features without too much computation time.

Motivated by its reasonable training time and the ability to use both high-dimensional and global features, our framework is based on the stacked learning method *Wolpert* (1992),

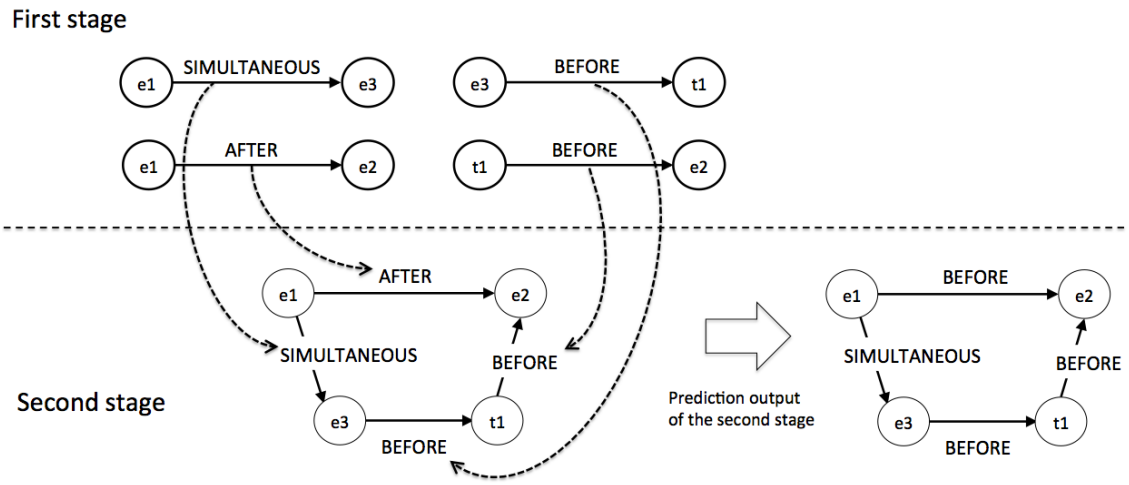


Figure 5.4: Stacked learning. The output from the first stage is treated as features for the second stage. The final output is predicted using label information of nearby TLINKs.

which employs two stages of classification as illustrated in Figure 5.4. The first stage employs a local model which predicts each TLINK independently. In the second stage, the relation types of neighbouring TLINKs, which are the output from the first stage, are used as non-local features. Since, in the test stage, the labels in the timegraphs are drawn from the predictions made by the local model, the training data for the second stage is created by using cross-validation techniques over the original data, and replacing the true labels with the predicted labels.

5.1.1 Non-local model

In the second stage, the document's timegraph is constructed and the output from the first stage is associated with TLINKs in the graph.

The classifiers in the second stage use the information from the nearby TLINKs and predict the final output. We exploit features extracted from the documents' timegraphs, as listed in Table 5.2 and 5.2 in the second stage of the stacked learning. A concrete example can be found in Figure 5.5 and the example column of Table 5.2 and 5.2 when the features of the relation between *Src* and *Dst* nodes are being extracted.

The timegraph features include information extracted from adjacent nodes and links, other paths that connect the temporal entities of the TLINK being predicted, generalized version of the paths, and tuples of edges and vertices along the paths. We treat timegraphs as directed graphs and double the number of edges by adding new edges with opposite relation types/directions to every existing edge. For example, if the graph contains an edge

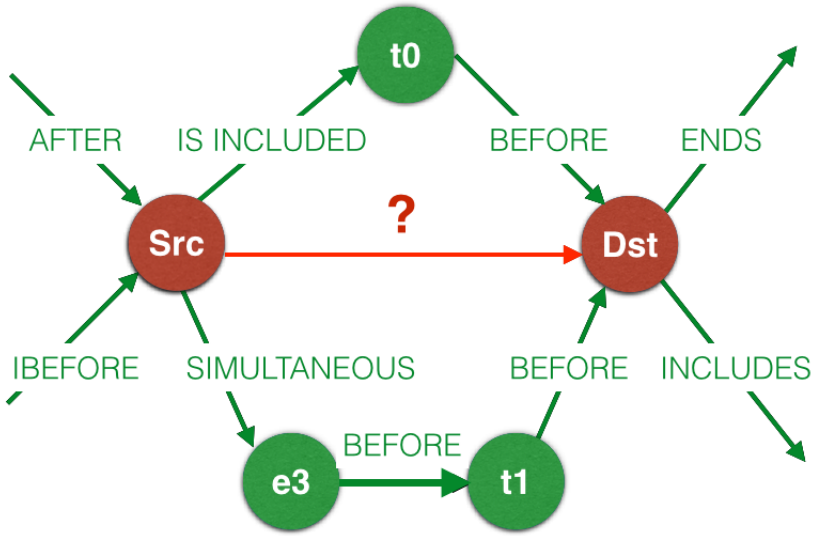


Figure 5.5: Prediction of the relation between *Src* and *Dst* nodes

e1_*BEFORE*_***e2***, we add a new edge ***e2***_*AFTER*_***e1***.

An example of a document's timegraph, which is constructed from the TLINKs in Figure 4.1, is shown in Figure 5.1.

5.2 Relation inference and time-time connection

We call TLINKs that have more than one path between the temporal entities *multi-path TLINKs*. The coverage of the multi-path TLINKs is presented in Table 5.4 and Figure 5.7. The original Timebank corpus has 4,983 TLINKs in total but only 282 of them have multiple paths between given pairs of temporal entities in the timegraphs. As we can see from the table that only 5.65% of all the annotated TLINKs are multi-path TLINKs, the annotated entities in the Timebank corpus create loosely connected timegraphs.

Since most of the timegraph features are only applicable for multi-path TLINKs, it is important to have dense timegraphs. To alleviate the sparsity problem of timegraphs, the relation inference in Subsection 5.2.1 and the time-time connection in Subsection 5.2.2 are performed in order to increase the timegraphs' density.

5.2.1 Relation inference

On the basis of the transitivity table proposed by *Allen* (1983), we construct a full set of inference relations as presented in Table B.1 and B.2. The table shows the results of

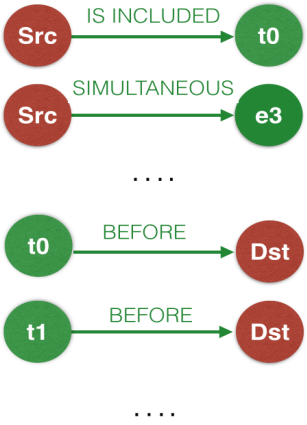
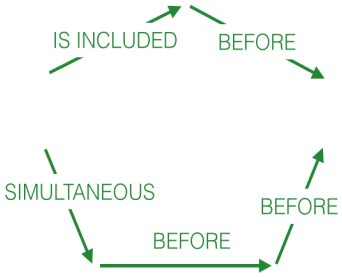
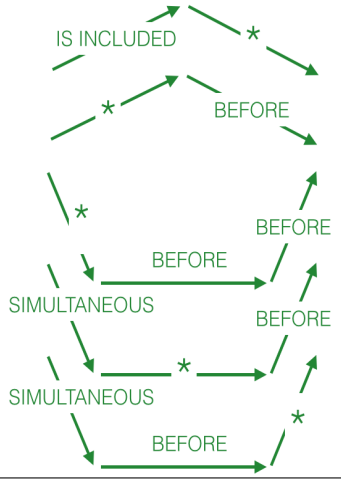
Feature	Example	Description
Adjacent nodes and links (ADJ)		The features are the concatenation of the directions to the adjacent links to the pair of entities, the relation types of the links, and the information on the adjacent nodes, i.e., word tokens, part of speech tags, lemmas. For example, *SRC_OUT-IS_INCLUDED-*(type of ***t0***), *DEST_IN-BEFORE-*(type of ***t0***).
Other paths (OP)		Paths with certain path lengths (in this work, 2 $\leq$ path length $\leq$ 4) between the temporal entities are used as features. The paths must not contain cycles. For example, the path features of the relation between ***e1*** and ***e2*** are *IS_INCLUDED-BEFORE* and *SIMULTANEOUS-SIMULTANEOUS-BEFORE*.
Generalized paths (GP)		The generalized version of the path features, e.g., the *IS_INCLUDED-BEFORE* path is generalized to **-BEFORE* and *IS_INCLUDED-**.

Table 5.2: Timegraph features (part 1)

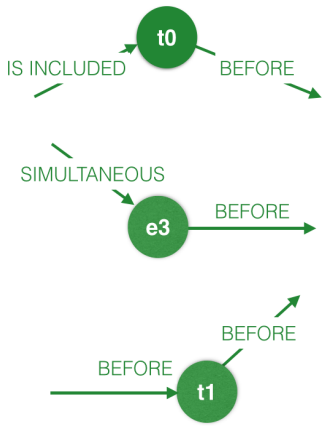
Feature	Example	Description
(E,V,E) tuples (EVE)		The (E,V,E) tuples of the edges and vertices on the path are used as features, e.g., *IS_INCLUDED_*(Type of ***t0***)*_BEFORE*. In this work, we use Type (for temporal expressions) and Tense, POS tags (for events) but others attributes could also be used.
(V,E,V) tuples (VEV)		The (V,E,V) tuples of the edges and vertices on the path are used as features, e.g., (Tense of ***e1***)*_IS INCLUDED_*(Type of ***t0***) and (Type of ***t0***)*_BEFORE_*(Tense of ***e2***).

Table 5.3: Timegraph features (part 2)

No. of TLINKs	E-E	E-T	Total
All TLINKs	2,520	2,463	4,983
Multi-path TLINKs	119	163	282
Percentage	4.72	6.62	5.65

Table 5.4: Coverage of multi-path TLINKs

temporal relation inference for every case, i.e., if a temporal entity A has relation X to an entity B and the entity B has relation Y to an entity C, then A has relation Z to C. We create new E-E and E-T connections between entities in a timegraph by following this set of inference rules. For example, if ***e1*** happens *AFTER* ***e2*** and ***e2*** happens *IMMEDIATELY_AFTER* ***e3***, then we infer a new temporal relation "***e1*** happens *AFTER* ***e3***".

In this study, we perform two types of inference: partial inference and full inference. For the partial inference, we add a new connection only when the inference gives only one type of temporal relation as a result from the relation inference. Full inference adds new connections for all of the inference results in Table B.1 and B.2.

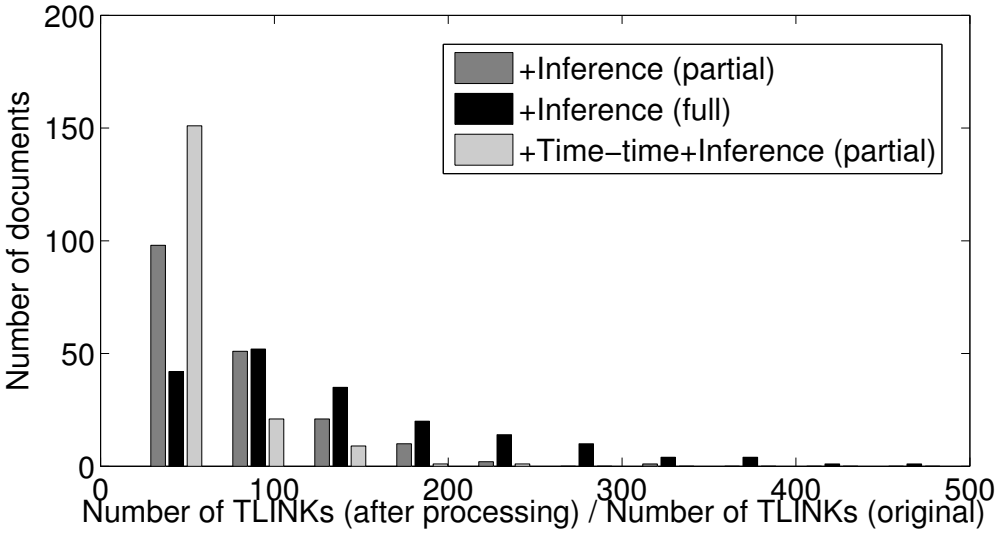


Figure 5.6: Histogram of the increased number of TLINKs for each document (The x-axis shows the ratio between the number of TLINKs after pre-processing and the original number of TLINKs. The y-axis is the number of documents.)

No. of TLINKs	**Total**
Original (annotated TLINKs)	4,983
+Inference (partial)	29,039
+Inference (full)	47,927
+Time-time connection +Inference (partial)	92,181

Table 5.5: Number of relations in Timebank

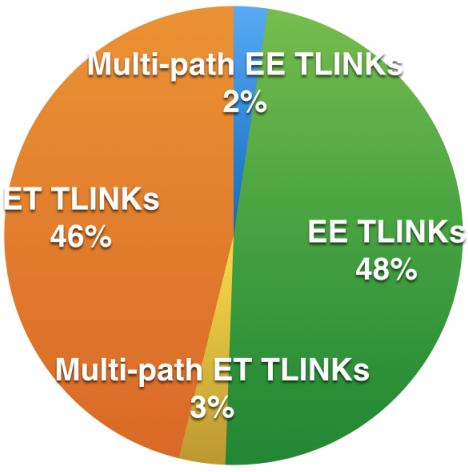


Figure 5.7: Visualization of coverage of multi-path TLINKs

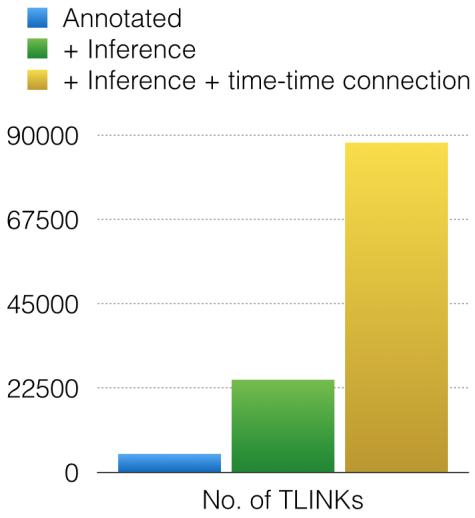


Figure 5.8: Visualization of number of TLINKs after relation inference and time-time connection

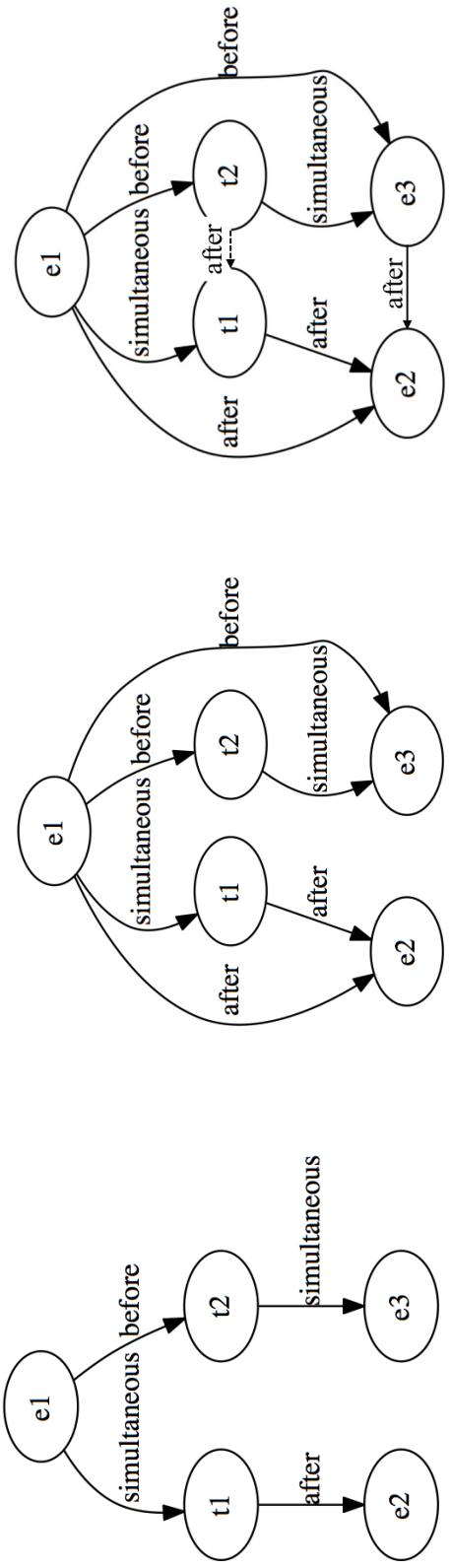


Figure 5.9: Relation inference and time-time connection. (a) Original timegraph. (b) After relation inference. Two relations (e1-e2, e1-e3) are added. (c) After time-time connection (t1-t2) and relation inference. Three relations (e1-e2, e1-e3, e2-e3) are added.

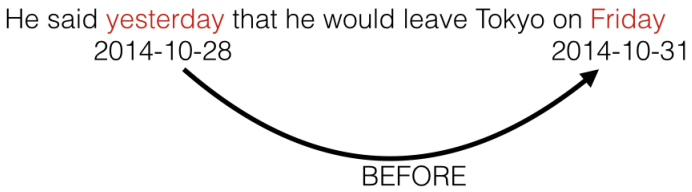


Figure 5.10: Time-time connection

Figure 5.9b shows a simple example of the timegraph after adding new inference relations to the original timegraph in Figure 5.9a. For example, knowing that the relation ***e1-t1*** is *SIMULTANEOUS* and ***t1-e2*** is *AFTER*, it can be inferred that the new temporal relation ***e1-e2*** is *AFTER*.

5.2.2 Time-time connection

As with *Chambers et al.* (2007) and *Tatu and Srikanth* (2008), we also create new connections between time entities in a timegraph by applying some rules to normalized values of time entities provided in the corpus. An example of how to add a time -time relation can be fount in Figure 5.10.

Figure 5.9c shows the timegraph after adding a time-time link and new inference relations to the original timegraph in Figure 5.9a. When the normalized value of ***t2*** is more than the value of ***t1***, a TLINK with the relation type *AFTER* is added between them. After that, as introduced in Subsection 5.2.1, new inference relations (***e1-e2***, ***e1-e3***, ***e2-e3***) are added.

As the number of relations grows too large and the computation time increases drastically after performing time-time connection and inference relation recursively, we limit the number of TLINKs for each document's timegraph to 10,000 relations. The total number of TLINKs for all documents in the corpus is presented in Table 5.5 and Figure 5.8. The first row is the number of the human-annotated relations. The second and third rows show the total number after performing relation inference and time-time connection. Table 7.1 shows the number of TLINKs for each relation type. The number of TLINKs after inference for some relation types, e.g, *IBEFORE*, *IAFTER*, *BEGINS*, *BEGUN BY*, and *ENDED BY*, decreases from the original number since the inference is performed over the predicted results made by the first stage of the stacking framework.

An example of a timegraph of a document before and after relation inference and time-time connection can be found in Figure 5.11.

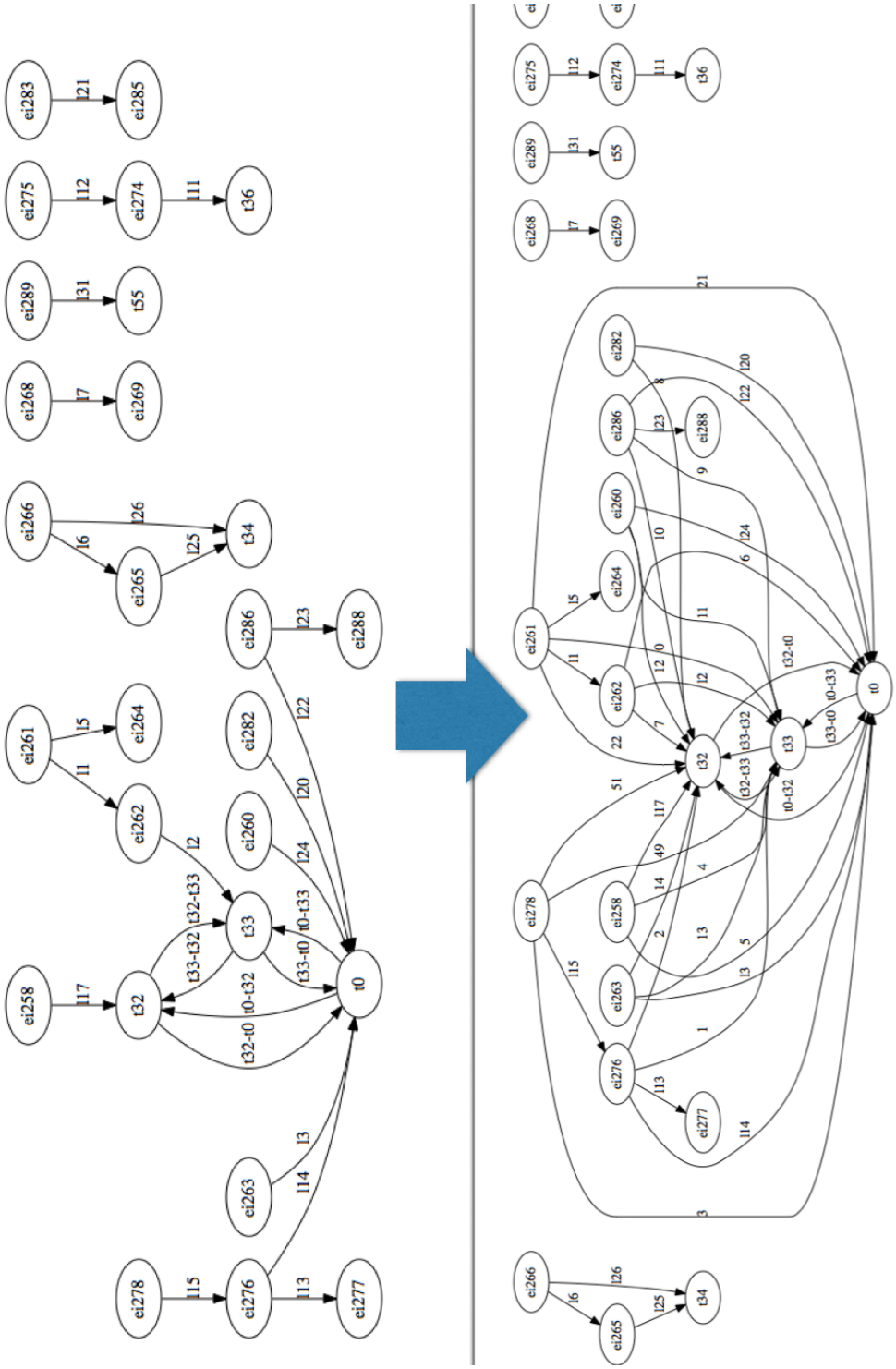


Figure 5.11: Example of a timegraph of a document before and after relation inference and time-time connection

Relation type	Original	Inference (partial)	Inference (full)	Time-time + Inference (partial)
BEFORE	1,190	9,061	13,682	36,223
AFTER	624	9,411	21,286	38,781
IBEFORE	24	3	19	6
IAFTER	31	3	12	6
INCLUDES	341	2,445	2,945	4,943
IS INCLUDED	1,227	2,435	3,171	4,950
BEGINS	42	7	21	10
BEGUN BY	49	8	18	10
ENDS	75	82	92	106
ENDED BY	85	81	85	104
SIMULTANEOUS	1,052	5,156	5,110	6,439
DURING	243	347	1,486	603

Table 5.6: Number of TLINKs for each relation type after relation inference and time-time connection

Figure 5.6 shows the distribution of how many times the number of TLINKs for each document increases after performing time-time connection and relation inference. As we can see from the histogram, the number of TLINKs for most of the documents' timegraphs increases by a factor of 50 compared to the original ones and the preprocessing can raise the number of TLINKs for a few documents by a factor of 200-500.

5.3 Evaluation

For the baselines and both stages of the stacked learning, we have used the LIBLINEAR invented by *Fan et al.* (2008) and configured it to work as L2-regularized logistic regression classifiers.

We trained our models on the Timebank corpus, introduced in Subsection 3.1.4, which was provided by the TempEval-3 organiser. The corpus contains 183 newswire articles in total.

5.3.1 Results on the training data

The performance analysis is performed based on 10-fold cross validation over the training data. The classification F1 score improves by 0.18% and 0.16% compared to the local pairwise models with/without deep syntactic features.

Approach	Graph-based evaluation		
	F1	P	R
Local - baseline features	58.15	58.17	58.13
Local - baseline + deep features	59.45	59.48	59.42
Stacked - baseline features	58.33	58.37	58.29
Stacked (inference) - baseline features	58.30	58.32	58.27
Stacked (inference, time-time) - baseline features	58.29	58.31	58.27
Stacked - baseline + deep features	59.55	59.51	59.58
Stacked (inference) - baseline + deep features	59.55	59.57	59.52
Stacked (inference, time-time) - baseline + deep features	**59.61**	**59.63**	**59.58**

Table 5.7: Ten-fold cross validation results on the training set

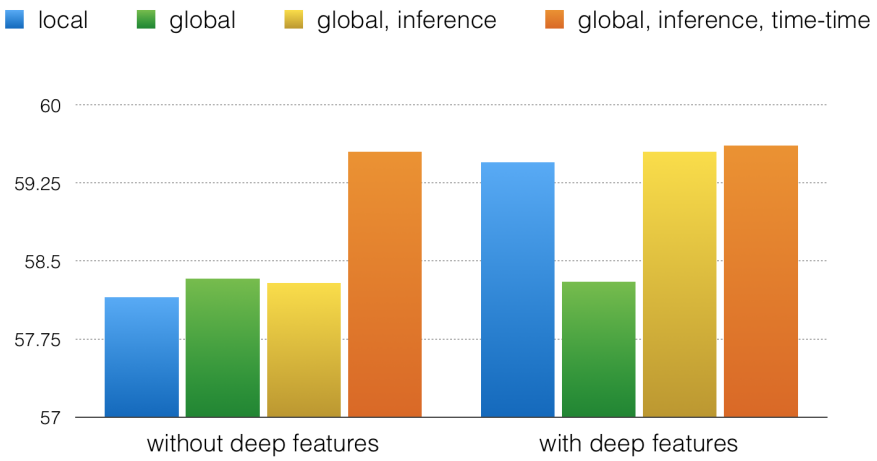


Figure 5.12: Visualization of results on the training data

We evaluated the system using a graph-based evaluation metric proposed by *UzZaman and Allen* (2011). Table 6.5 shows the classification accuracy over the training set using graph-based evaluation and the visualization of the result is presented in Figure5.12.

Further experimental results can be found in *Laokulrat et al.* (2014).

VI

Using probability values as real-valued features

In this chapter, the probability values of the prediction in the first stage are applied in temporal relation classification as real-valued features.

6.1 Probability values as real-valued features

For the first stage (local model) of the stack learning, we use logistic regression classifiers, so it gives not only the output relation but also the probability values (confidence scores) of all relation types. We make use of these probability values obtained from the first stage as real-valued features in the second stage (global model) of the classifier. The features that are used in the local models for E-E and E-T classifiers can be found in Table 6.2 and 6.3.

Table 6.1 describes the real-valued features that we construct from the output relation types and their probability values. The example of the features is illustrated in Figure 6.1. In Figure 6.1a, the first stage predicts the temporal relations of ***e1-e3*** and ***e3-t1*** and gives the probability values of each relation type. We create the timegraph features in Figure 6.1b by generating all combinations of the relation types and calculating the real-valued features by multiplying the probability values. All possible inference results of the features in Figure 6.1b are also used as timegraph features as presented in Figure 6.1c. If the inference result is *UNDEFINED*, the feature will not be used.

Feature	Description
Probability values of the output	The probability values of the predicted output from the first stage. All of the output labels that have more probability values than a specified threshold are used as real-valued features. For example, for predicting in the relation of ***e1*** and ***e3*** shown in Figure 6.1a in the second stage of the stacked learning, the features include *BEFORE:0.1*, *AFTER:0.05*, *IBEFORE:0.2*, and so on. Refer to Figure 6.1a.
All possible paths	All possible paths between the pair of temporal entities are used as features. For example, the path features of the relation between ***e1*** and ***t1*** are *BEFORE-BEFORE:0.04*, *BEFORE-AFTER:0.001*, and so on. Refer to Figure 6.1b.
Inference results of all possible paths	Inference results of all possible paths between the pair of temporal entities are used as features. For example, the inference results of the relation between ***e1*** and ***t1*** are *BEFORE:0.04*, *BEFORE:0.3* (summed up to *BEFORE:0.34*), *ENDS:0.0007*, *ENDED BY:0.0007*, and so on. Refer to Figure 6.1c.

Table 6.1: Probability values as real-valued features

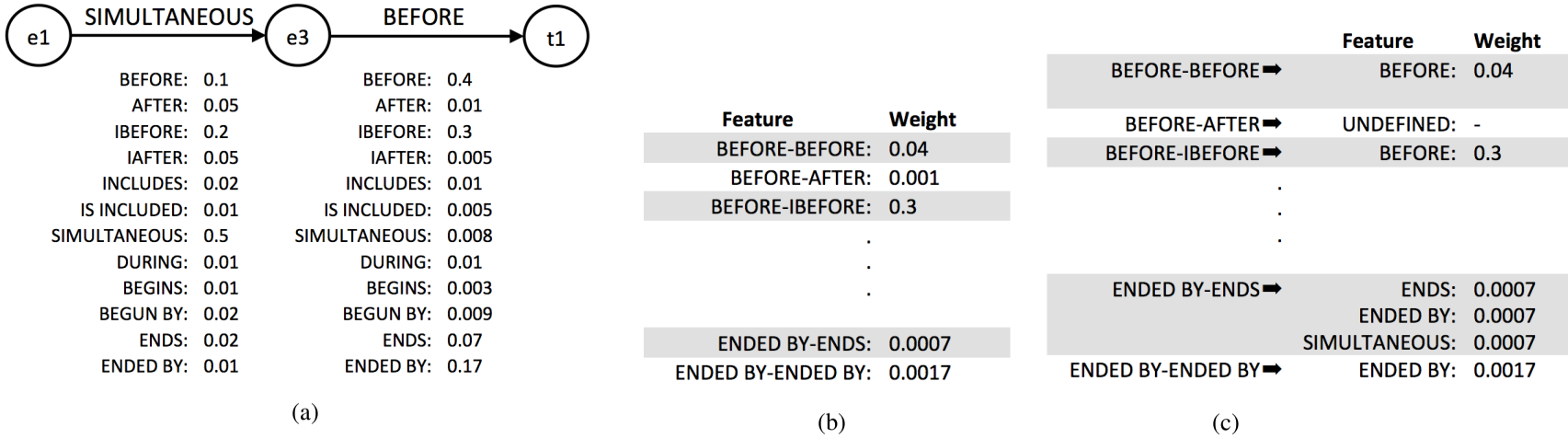


Figure 6.1: Using probability values as real-valued features. (a) Probability values of the output from the first stage. (b) All possible paths with the probability values as real-valued features. (c) All possible inference results from the possible paths.

6.2 Experimental evaluation and results

For the baselines and both stages of the stacked learning, we have used the LIBLINEAR, invented by *Fan et al.* (2008), and configured it to work as L2-regularized logistic regression classifiers.

We trained our models on the Timebank corpus, introduced in Subsection 3.1.4, which was provided by the TempEval-3 organizer. The corpus contains 183 newswire articles in total.

We performed the experiments using various combinations of features and configurations. For cross-validation evaluation over the training data, we performed a grid search in order to find the best regularization coefficient for each model. In each model, we used the same regularization coefficient value for every fold. We also applied that value in the test stage. Refer to Table 6.4 for the details of each configuration.

We performed the relation inference in both training and testing stages in order to increase timegraphs' density. However, we did not count the propagated relations as evaluation relations. We evaluated only the relations annotated in the original corpus.

Feature	E-E	E-T	Description
Event attributes			
Class	X	X	All attributes associated with events. The explanation of each attribute can be found in *Pustejovsky et al.* (2005).
Tense	X	X	
Aspect	X	X	
Modality	X	X	
Polarity	X	X	
Timex attributes			
Type		X	All attributes associated with temporal expressions. The explanation of each attribute can be found in *Pustejovsky et al.* (2005).
Value		X	
FunctionInDocument		X	
TemporalFunction		X	
Morphosyntactic information			
Words	X	X	Words, POS, lemmas within a window before/after event words extracted using Stanford coreNLP invented by *Manning et al.* (2014)
Part of speech tags	X	X	
Lemmas	X	X	
Lexical semantic information			
Synonyms of event word tokens	X	X	WordNet lexical database invented by *Fellbaum* (2010)
Synonyms of temporal expressions		X	
Event-Event information			
Class match	X		Details are described in *Chambers et al.* (2007)
Tense match	X		
Aspect match	X		
Class bigram	X		
Tense bigram	X		
Aspect bigram	X		
Same sentence	X	X	True if both temporal entities are in the same sentence
Deep syntactic information			
Phrase structure	X	X	Deep syntactic information extracted from Enju Parser invented by *Miyao and Tsujii* (2008). The details are described in *Laokulrat et al.* (2013b)
Predicate-argument structure	X	X	

Table 6.2: Local (baseline + deep) feature usage

Feature	E-E	E-T	Description
Adjacent nodes and links	X	X	
Other paths	X	X	The details are described in Subsection 5.1
Generalized paths	X	X	
(E,V,E) tuples	X	X	
(V,E,V) tuples	X	X	

Table 6.3: Timegraph feature usage

Configuration	Description
A1	local (local model)
B1	local + graph (stack model)
C1	local + graph (stack model, partial inference)
D1	local + graph (stack model, time-time relations, partial inference)
E1	local + graph (stack model, full inference)
F1	local + graph + probability (stack model, partial inference)
G1	local + graph + probability (stack model, time-time relations, partial inference)
A2	local + deep (local model)
B2	local + deep + graph (stack model)
C2	local + deep + graph (stack model, partial inference)
D2	local + deep + graph (stack model, time-time relations, partial inference)
E2	local + deep + graph (stack model, full inference)
F2	local + deep + graph + probability (stack model, partial inference)
G2	local + deep + graph + probability (stack model, time-time relations, partial inference)

Table 6.4: Combination of features and configurations

	Graph-based evaluation			Accuracy (%)	
	F1 (%)	P (%)	R (%)	E-E	E-T
A1	58.07	58.04	58.09	48.21	67.11
B1	58.35	58.39	58.31	48.29	67.52
C1	58.12	58.12	58.11	48.01	67.28
D1	58.43	58.44	58.41	48.57	67.11
E1	58.15	58.17	58.13	48.25	67.19
F1	58.27	58.23	58.31	48.57	67.07
G1	58.36	58.31	58.41	48.45	67.07
A2	59.35	59.36	59.34	50.12	67.52
B2	59.62	59.61	59.62	50.40	**67.80**
C2	59.52	59.57	59.46	50.63	67.68
D2	59.60	59.63	59.56	50.71	67.52
E2	59.25	59.31	59.18	50.04	67.60
F2	**59.69**	**59.68**	**59.71**	**51.31**	67.52
G2	59.62	59.60	59.64	**51.31**	67.15

Table 6.5: 10-fold cross validation results on the training data when all the features are used. Refer to Table 6.4 for the details of each configuration setting.

6.2.1 Results on the training data

The performance analysis is performed based on 10-fold cross validation over the training data[1]. We evaluated the system using both pairwise accuracy and the graph-based evaluation metric proposed by *UzZaman and Allen* (2011). Table 6.5 shows the classification results over the training set using graph-based evaluation and the accuracy of the E-E and E-T models separately.

Based on the graph-based evaluation, the classification F1 score improves by 0.36 *pp* (difference between A1 and D1) and 0.34 *pp* (difference between A2 and F2) compared to the local models with/without deep syntactic features. However, by looking at the accuracy of the E-E and E-T classification results, we can see that the stacked method helped the prediction in the E-E models but decreased the accuracy of the E-T models.

Table 6.7 shows the top F1 scores of different set of graph features and configuration. In every model shown in this table, both local and deep features are used. The results ranked 1^{st} - 20^{th} in the table used the stacked models and the bottom row shows the result of the local model. The definition of the acronyms of each feature set can be found in Table 5.2 and 5.3. From the table, the stacked models with the probability values outperformed other

[1]We performed the document-level 10-fold cross validation. The folds were generated by: (1) Sorting the documents in ascending order of file names. (2) Selecting the 1^{st}, 11^{th}, 21^{st}, ... documents to be the test data of the 1^{st} fold and the others are the training data. (3) Selecting the 2^{nd}, 12^{th}, 22^{nd}, ... documents to be the test data of the 2^{nd} fold and the others are the training data, and so on.

	Graph-based evaluation			Accuracy (%)	
	F1 (%)	**P (%)**	**R (%)**	**E-E**	**E-T**
A1	52.86	52.26	53.48	41.64	**73.26**
B1	53.80	53.23	54.39	43.15	71.27
C1	53.81	53.58	54.05	42.86	70.66
D1	52.64	52.27	53.02	44.82	66.56
E1	52.10	51.54	52.68	41.55	71.44
F1	53.06	52.75	53.36	43.69	69.82
G1	52.46	51.91	53.02	44.75	67.29
A2	52.33	51.76	52.91	42.38	71.74
B2	52.81	52.38	53.25	43.43	70.81
C2	**54.70**	**54.22**	**55.19**	**45.76**	69.85
D2	53.29	52.56	54.05	44.64	69.20
E2	51.62	51.14	52.11	41.54	71.32
F2	53.53	52.92	54.16	45.22	70.17
G2	52.86	52.14	53.59	45.14	68.08

Table 6.6: Results on the test data when all the features are used. Refer to Table 6.4 for the details of each configuration setting.

configuration as we can see that every model that ranked in the 20 highest F1 scores used the probability values as real-valued features. The best model without probability values ranked 41^{st} in our feature experiment which proves that the probability values are helpful.

Compared to the local model with deep features (baseline), the stacked model with the highest F1 score (60.25%) changed the relation types of 259 (out of 2520) E-E TLINKs and 173 (out of 2463) E-T TLINKs. The overall improvement is statistically significant* ($p < 0.01$, McNemar's test, two-tailed) when applying the best feature set and configuration to the system.

Rank	Feature set	Inference	Time-Time	Probability	F1 (%)	P (%)	R (%)
1	OP (length = 2, 3) + GP + VEV	x	x	x	**60.25**	**60.20**	**60.29**
2	OP (length = 2, 3) + GP + EVE + VEV	x	x	x	60.22	60.18	60.13
3	OP (length = 2, 3) + EVE + VEV	x	x	x	60.08	60.04	60.27
4	OP (length = 2, 3)	x	x	x	60.06	60.04	60.09
5	OP (length = 2, 3) + VEV	x	x	x	60.04	60.04	60.09
6	OP (length = 2)	x	x	x	60.03	59.98	60.09
7	OP (length = 2, 3, 4)	x	x	x	60.03	59.98	60.09
8	OP (length = 2, 3) + GP + EVE	x		x	59.98	59.94	60.03
9	OP (length = 2) + GP	x	x	x	59.98	59.98	59.99
10	OP (length = 2, 3, 4) + GP + VEV	x	x	x	59.98	59.93	60.03
11	OP (length = 2, 3, 4) + GP + EVE + VEV	x		x	59.97	59.92	60.03
12	OP (length = 2, 3, 4) + GP	x		x	59.97	59.92	60.01
13	OP (length = 2) + GP + EVE + VEV	x	x	x	59.96	59.97	59.95
14	OP (length = 2, 3) + GP	x	x	x	59.95	59.94	59.97
15	OP (length = 2, 3, 4) + GP + EVE + VEV	x		x	59.94	59.92	59.97
16	OP (length = 2, 3, 4) + GP + EVE + VEV	x	x	x	59.93	59.90	59.97
17	OP (length = 2) + GP + VEV	x	x	x	59.93	59.94	59.93
18	OP (length = 2, 3) + GP	x		x	59.93	59.87	59.99
19	OP (length = 2, 3)	x		x	59.93	59.89	59.97
20	OP (length = 2, 3) + GP + EVE	x	x	x	59.92	59.91	59.93
	Local + deep features				59.35	59.36	59.34

Table 6.7: 20 highest F1 scores of different selected sets of graph features. Local and deep features are used in every model. The scores were obtained by performing 10-fold cross validation over the training data.

System	F1 (%)	P (%)	R (%)
UTTime	56.45	55.58	57.35
NavyTime	46.83	46.59	47.07
JU-CSE	34.77	35.07	34.48
Our system (F2, All feat.)	57.30	57.01	57.58
Our system (G2, Best feat.)	**57.78**	**57.63**	**57.92**

Table 6.8: Comparison to other systems submitted to TempEval-3. The TempEval-3 test data set was used.

6.2.2 Results on the test data

Table 6.6 shows the results on the test data containing 20 newswire articles, which were manually annotated and provided by the TempEval-3 organizer. The stacked models with partial inference (C1, C2) improve the classification F1 score by 0.95 *pp* (difference between A1 and C1) and 2.37 *pp* (difference between A2 and C2) compared to the baselines with/without deep syntactic features. However, this is not statistically significant ($p > 0.1$, McNemar's test, two-tailed). Besides, incorporating probability values into the model deteriorated the classification results (although this is, again, not statistically significant).

As shown in Table 6.8, we also compared our system to other systems that were submitted to TempEval-3's task C-relation-only. In order to make a fair comparison, the models were trained on the Timebank and AQUAINT corpora, containing 256 newswire articles in total, which were provided by TempEval-3's organizer. We do not use the AQUAINT corpus in cross-validation evaluation because there are many duplications in the data set so we think it will make the results unreliable.

Note that all the TempEval-3 participants only use local information for temporal relation classification and our system, UTTime proposed in *Laokulrat et al.* (2013b), utilizes deep syntactic features. The system description for NavyTime and JU-CSE can be found in *Chambers* (2013) and *Kolya et al.* (2013).

We can see that the global model (F2) can achieve a better results. By applying all of the features to the model, it reaches an F1 score of 57.30 %. We also did the experiment using the selected set of features that got the highest F1 score in Table 6.7 and achieved an F1 score of 57.78 % which outperformed all other systems. The training times for the local model (A2) and the global models (F2, G2) were 16.89, 21.50, and 25.50 seconds respectively[2].

[2]The training times do not include feature pre-processing. The experiments were run on a 64-bit machine with Intel Core i7 1.8GHz CPU, and 4GB main memory.

	Total	Avg. no. of features		Total	Avg. no. of features
A1	31,895	63.31	A2	33,778	68.89
B1	32,923	87.69	B2	34,816	93.28
C1	34,321	127.38	C2	36,299	133.92
D1	35,294	218.64	D2	37,236	226.07
E1	36,559	188.53	E2	37,528	194.64
F1	36,374	395.79	F2	38,338	402.33
G1	37,347	487.05	G2	38,274	494.48

Table 6.9: Total number of features and average number of features for a TLINK. Refer to Table 6.4 for the details of each configuration setting.

Approach	E-E
Chambers08 (local)	66.8
Chambers08 (global)	70.4
Our system (A1)	69.7
Our system (B1)	69.2
Our system (C1)	69.0
Our system (D1)	69.6
Our system (F1)	69.6
Our system (G1)	**70.1**
Our system (A2)	70.8
Our system (B2)	71.0
Our system (C2)	70.8
Our system (D2)	70.8
Our system (F2)	71.9
Our system (G2)	**72.2**

Table 6.10: Comparison with Chambers's system (Accuracy(%)) by performing 10-fold cross validation over the Timebank corpus. The configuration of our system is described in Table 6.4.

6.2.3 Comparison with the state of the art

We compared our system to those of *Chambers and Jurafsky* (2008) and *Yoshikawa et al.* (2009), which use global information to improve the accuracy of temporal relation classification.

In *Chambers and Jurafsky* (2008), the experiments were performed on the Timebank corpus over the relations *BEFORE* and *AFTER*. They merged *IBEFORE* and *IAFTER* into those relations as well and ignored all other relations. Table 6.10 shows our results when using the same experiment setting. Although the difference when using the local model and the stacked model is not significant, the overall scores of our system were better than those of their system.

Approach	Task A	Task B	Task C	Overall
Yoshikawa09 (local)	61.3	78.9	53.3	66.7
Yoshikawa09 (global)	66.2	79.9	55.2	68.9
Our system (A1)	59.7	**80.2**	58.7	68.5
Our system (B1)	60.1	79.9	58.7	68.5
Our system (C1)	59.1	79.4	58.8	68.0
Our system (D1)	61.8	79.9	59.8	69.2
Our system (F1)	63.0	79.5	59.4	69.2
Our system (G1)	**63.3**	79.9	**59.9**	**69.6***
Our system (A2)	62.3	**80.2**	58.5	69.1
Our system (B2)	63.6	79.8	58.7	69.3
Our system (C2)	62.4	79.6	58.7	68.9
Our system (D2)	64.4	80.0	58.9	69.7
Our system (F2)	64.4	79.6	58.9	69.4
Our system (G2)	**64.8**	80.0	**59.1**	**69.8****

Table 6.11: Comparison with Yoshikawa's system (Accuracy(%)) by performing 10-fold cross validation over the TempEval-07 training data. The configuration of our system is described in Table 6.4.

We compared our system to that of *Yoshikawa et al.* (2009) which was evaluated based on TempEval-07's rules and data set organized by *Verhagen et al.* (2007), in which the relation types were reduced to six relations: *BEFORE*, *OVERLAP*, *AFTER*, *BEFORE-OR-OVERLAP*, *OVERLAP-OR-AFTER*, and *VAGUE*. The evaluation was done using 10-fold cross validation over the same data set as that of their reported results.

According to TempEval-07's rules, there are three tasks as follows:

- Task A: Temporal relations between events and all time expressions appearing in the same sentence.
- Task B: Temporal relations between events and the DCT.
- Task C: Temporal relations betweeen main verbs of adjacent sentences.

As shown in Table 6.11, our system can achieve better results in task B and C even without deep syntactic features but performs worse than their system in task A. Compared to the baselines, the overall improvement is statistically significant* ($p < 0.05$, McNemar's test, two-tailed) without deep syntactic features and gets more statistically significant** ($p < 0.01$, McNemar's test, two-tailed) when applying deep syntactic information to the system. The overall result has about 0.9 *pp* (difference between Yoshikawa09 (global) and

G2) higher accuracy than the result from their global model. Note that *Yoshikawa et al.* (2009) did not apply deep syntactic features in their system.

The stacked model enhances the classification accuracy of task A when timegraphs are dense enough. Deep syntactic features can be extracted only when temporal entities are in the same sentences so they improve the model for task A (event-time pairs in the same sentences) but these features clearly lower the accuracy of task C, since there are very few event-event pairs that appear in the same sentences (and break the definition of task C). This is probably because the sparseness of the deep features degrades the performance in task C. Moreover, these features do not help task B in the local model because we cannot extract any deep syntactic features from TLINKs between events and DCT. However, they contribute slightly to the improvement in the stacked model since deep syntactic features increase the accuracy of the prediction of task A in the first stage of the stacked model. As a result, timegraph features extracted from the output of the first stage are better than those extracted from the local model trained on only baseline features.

VII

Discussion

As we can see from Table 7.1, the distribution of the training data is very biased. Some relation types appear in the corpus less than 50 times while some of them appear more than 500 times. This is probably one of the major reasons that we could not obtain a high classification performance.

According to the results shown in Table 4.2, the predicate-argument-structure features contributed to the improvement more than those of phrase structures in both precision and recall. The reason is probably that the features from phrase structures that we used did not imply a temporal relation of events in the sentence. For instance, the two following sentences give exactly the same path of the event words in the phase structure trees shown in Figure 7.1 and 7.2 .

John **saw** *Mary before the* **meeting.**

John **saw** *Mary after the* **meeting.**

Using semantic structures, such as predicate-argument structure, can improve the performance. Besides, using different methods of extracting features from phase structure trees may also give better results.

As we can see from Table 6.5 and 6.6, although deep syntactic features can improve the classification accuracy significantly, some additional pre-processing is required. Moreover, deep parsers are not able to parse sentences accurately in some specific domains. Thus, sometimes it is not practical to use this kind of features in real-world temporal relation classification problems. By applying the stacked learning approach to the temporal relation classification task, the system with only local features is able to achieve good classification results compared to the system with deep syntactic features. The stacked model also has another advantage that it is easy to build and does not consume too much training time compared to MLNs used by *Yoshikawa et al.* (2009), which are, in general, computationally expensive and infeasible for large training sets.

Relation type	Number of occurence
BEFORE	1,190
AFTER	624
IBEFORE	24
IAFTER	31
INCLUDES	341
IS INCLUDED	1,227
BEGINS	42
BEGUN BY	49
ENDS	75
ENDED BY	85
SIMULTANEOUS	1,052
DURING	243

Table 7.1: Number of TLINKs for each relation type

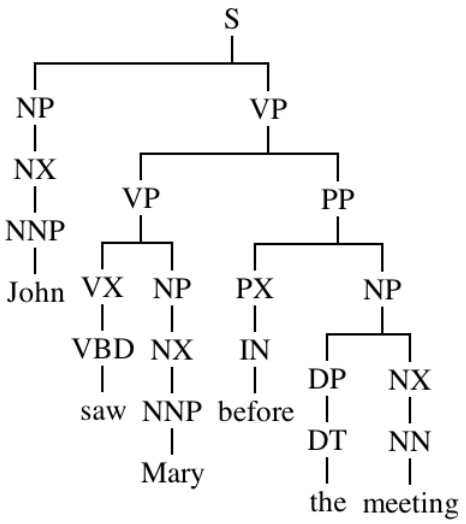


Figure 7.1: Phase structure tree for the sentence *John saw Mary before the meeting*

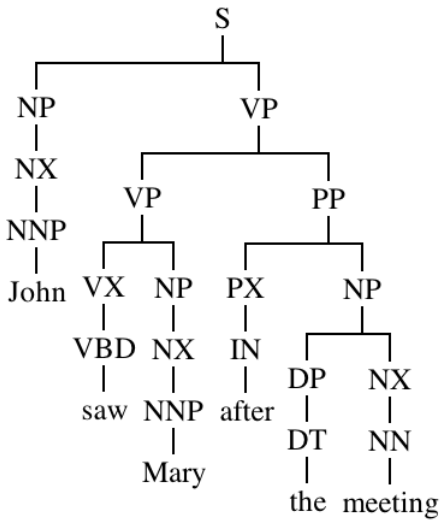


Figure 7.2: Phase structure tree for the sentence *John saw Mary after the meeting*

Again, from Table 6.5 and 6.6, the inference and time-time connection, described in Section 5.2.1, sometimes degrade the performance. This is presumably because the number of features increases severely as the number of TLINKs increased. Table 6.9 shows the total number of features and the average number of features used for one TLINK for each configuration. However, when the adjacent nodes and edges are removed from the feature set, the inference (partial) and time-time connection play an important role in the classification improvement as we can see in Table 6.7 that these steps were performed in all of the systems with top F1 scores. The adjacent nodes and edges decrease the accuracy since they increase the number of features significantly, especially when the relation inference and time-time connection are performed, so excluding those features makes a significant improvement.

Full relation inference is not helpful and also decreases the accuracy. This is probably because the severe increase of the *'adjacent nodes and edges'* features when adding too many TLINKs to the timegraphs. Having too many features also causes the sparseness problem. As we can see from Table 6.7 that only graph paths with path length = 2, 3 are useful while including paths with length = 4 into the models decreases the accuracy.

Relation type	No. of TLINKs	A1	A2	B2	C2	D2	E2	F2	G2
BEFORE	490	40.91	47.85	44.85	45.17	46.63	45.15	48.47	**48.67**
AFTER	458	36.36	39.96	40.22	**40.85**	40.83	40.71	39.96	40.61
IBEFORE	22	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
IAFTER	27	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
INCLUDES	171	17.84	19.30	23.04	**23.11**	13.45	21.30	16.37	16.37
IS INCLUDED	212	18.46	13.68	21.66	21.51	15.57	**23.19**	17.45	20.28
BEGINS	24	0.0	**8.33**	0.0	0.0	0.0	0.0	**8.33**	4.17
BEGUN BY	24	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ENDS	12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ENDED BY	44	35.71	0.0	36.36	37.93	0.0	**39.29**	0.0	0.0
SIMULTANEOUS	990	64.78	77.47	65.82	66.42	**81.11**	66.01	80.00	79.39
DURING	46	0.0	2.17	0.0	**4.26**	0.0	**4.26**	2.17	2.17

Table 7.2: F1 score (%) of the prediction for E-E TLINKs. The scores were obtained by performing 10-fold cross validation over the training data. The details of each configuration are described in Table 6.4.

Relation type	No. of TLINKs	A1	A2	B2	C2	D2	E2	F2	G2
BEFORE	700	81.46	87.14	82.81	82.55	87.29	83.23	87.0	**88.0**
AFTER	166	54.62	45.18	58.96	56.72	46.99	**59.04**	45.18	45.18
IBEFORE	2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
IAFTER	4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
INCLUDES	170	51.27	54.12	51.59	53.82	54.71	51.53	**55.88**	**55.88**
IS INCLUDED	1,015	73.04	**82.56**	73.71	73.55	81.87	73.62	82.27	82.36
BEGINS	18	0.0	0.0	0.0	0.0	0.0	0.0	**11.11**	**11.11**
BEGUN BY	25	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ENDS	63	14.08	12.70	**25.29**	15.38	20.63	20.93	14.29	20.63
ENDED BY	41	0.0	12.50	20.41	20.41	12.50	**24.0**	15.00	12.50
SIMULTANEOUS	62	0.0	1.61	**5.71**	2.74	3.23	5.19	3.23	1.61
DURING	197	13.44	17.26	19.08	**21.02**	19.29	19.73	16.75	17.77

Table 7.3: F1 score (%) of the prediction for E-T TLINKs. The scores were obtained by performing 10-fold cross validation over the training data. The details of each configuration are described in Table 6.4.

Type	No. of TLINKs
E-E (SS)	1,592
E-E (DS)	928
E-T (SS)	1,296
E-T (DCT)	1,167
Total	4,983

Table 7.4: Number of TLINKs for each link type when using 4 classifiers

Based on the 10-fold cross validation of the training data, the classification F1 scores (pairwise evaluation) for each relation type are shown in Table 7.2 and 7.3. The relation types that have low number of occurrences in the training set, e.g. *IBEFORE*, *IAFTER*, *BEGINS*, *BEGUN_BY*, *ENDS*, and *ENDED_BY*, are almost 100% misclassified. Unfortunately, the F1 scores do not give any further insights on how each configuration of the stacked models affects the classification results.

We have also performed a more detailed evaluation by dividing TLINKs into 4 categories:

- Two events within the same sentence (E-E (SS)).
- Two events in adjacent sentences (E-E (DS)).
- An event and a temporal expressions within the same sentence (E-T (SS)).
- An event and the DCT (E-T (DCT)).

The number of TLINKs for each category is shown in Table 7.4. We trained the models on the Timebank corpus separately for each link type and performed a grid search to find the best regularization parameters for each model.

Table 7.5 shows the accuracy (%) of each classifier by performing 10-fold cross validation. The weighted average values, E-E and E-T, are comparable to the results in Table 6.5. The accuracy for E-E and E-T classification of F2, which is the best configuration, improved by 0.79 *pp* and 2.31 *pp* respectively (compared to F2 of Table 6.5). We also performed the graph-based evaluation on the F2 classification results and achieved the F1 score, precision, and recall measures of 61.63, 61.55, and 61.70, which are the best performance so far. The results in Table 7.5 also suggest that the deep syntactic information is more useful in classifying E-E (SS) TLINKs than in classifying E-T (SS) TLINKs. The timegraph features yield better results in E-E (SS) and E-E (DS) classifiers while E-T (SS)

	Accuracy (%)				Average	
	E-E (SS)	**E-E (DS)**	**E-T (SS)**	**E-T (DCT)**	**E-E**	**E-T**
A1	41.08	61.53	63.66	74.04	48.61	68.57
F1	41.71	**62.50**	63.89	73.86	49.37	68.62
G1	41.58	62.39	64.27	73.18	49.25	68.49
A2	45.79	61.53	**64.89**	74.04	51.59	69.22
F2	**46.48**	61.75	64.81	**75.41**	**52.10**	**69.83**
G2	46.04	61.74	64.66	74.12	51.82	69.02

Table 7.5: Accuracy (%) by performing 10-fold cross validation over the Timebank corpus. E-E shows the weighted average over E-E (SS) and E-E (DS). E-T shows the weighted average over E-T (SS) and E-T (DCT).

and E-T (DCT) classifiers have inconsistent results between training with and without deep syntactic information. The overall improvement (F2 over A2) by exploiting timegraph features when using deep syntactic information is statistically significant ($p < 0.001$, McNemar's test, two-tailed).

VIII

Conclusion

The system, UTTime, identifying temporal links and classifying temporal relation, is proposed. The links were identified based on the rule-based approach and then some links were filtered out by a classifier. The filtering helped improve the system considerably. For the relation classification task, the features extracted from phrase structures and predicate-argument structures were proposed, and the features improved the classification in precision, recall, and F-score.

In this study, a stacked model for the temporal relation classification task as well as the incorporation of non-local features and probability values are proposed. We also apply the relation inference rules and the time-time connection to tackle the timegraphs' sparseness problem. The evaluation has been carried out to confirm the effectiveness of the proposed method and the analysis of the effectiveness of each feature type has also been performed.

From the evaluation results, using probability values as real-valued features can improve the classification accuracy especially when selecting the best feature set and the best model configuration. Our system outperforms other systems that were submitted to TempEval-3 and achieve good accuracy even without applying deep syntactic features. The evaluation results also show that our system achieves the better results than those of the state-of-the-art systems.

In future work, we hope to do more analysis on the classification results, especially the effect of features on each relation type, and apply our method to other relation classification tasks.

APPENDICES

A

TimeML

This appendix shows an example of a full TimeML document (ABC19980304.1830.1636.tml) in The Timebank corpus which is annotated following the TimeML specification to indicate events, temporal expressions, and temporal relations.

```
<TimeML xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://timeml.org/timeMLdocs/TimeML_1.2.1.xsd">
  <DOCID> ABC19980304.1830.1636 </DOCID>
  <DCT>
    <TIMEX3 tid="t0" type="DATE" value="1998-03-04" temporalFunction="false" functionInDocument="CREATION_TIME"> 19980304 </TIMEX3>
  </DCT>
  <EXTRAINFO> ABC19980304.1830.1636 NEWS STORY </EXTRAINFO>
▼ <TEXT>
    Finally
    <TIMEX3 tid="t32" type="DATE" value="1998-03-04" temporalFunction="true" functionInDocument="NONE" anchorTimeID="t0"> today </TIMEX3>
    , we
    <EVENT eid="e1" class="REPORTING"> learned </EVENT>
    that the space agency has finally
    <EVENT eid="e2" class="OCCURRENCE"> taken </EVENT>
    a giant leap forward. Air Force Lieutenant Colonel Eileen Collins will be
    <EVENT eid="e3" class="I_ACTION"> named </EVENT>
    <EVENT eid="e37" class="STATE"> commander </EVENT>
    of the Space Shuttle Columbia for a
    <EVENT eid="e38" class="OCCURRENCE"> mission </EVENT>
    in
    <TIMEX3 tid="t33" type="DATE" value="1998-12" temporalFunction="true" functionInDocument="NONE" anchorTimeID="t0"> December </TIMEX3>
    . Colonel Collins has been the
    <EVENT eid="e40" class="STATE"> co-pilot </EVENT>
    before, but this time she's the
    <EVENT eid="e41" class="STATE"> boss </EVENT>
    . Here's ABC's Ned Potter. Even two hundred miles up in space, there has been a glass ceiling. It wasn't until
    <TIMEX3 tid="t34" type="DURATION" value="P20Y" temporalFunction="false" functionInDocument="NONE"> twenty years </TIMEX3>
    after the first astronauts were
    <EVENT eid="e6" class="OCCURRENCE"> chosen </EVENT>
    that NASA finally
    <EVENT eid="e7" class="OCCURRENCE"> included </EVENT>
    six women, and they were all scientists, not pilots. No woman has actually been
    <EVENT eid="e254" class="STATE"> in </EVENT>
    charge of a mission until
    <TIMEX3 tid="t55" type="DATE" value="PRESENT_REF" temporalFunction="true" functionInDocument="NONE" anchorTimeID="t0"> now </TIMEX3>
    . Just the fact that we're
    <EVENT eid="e68" class="OCCURRENCE"> doing </EVENT>
    the job that we're doing
    <EVENT eid="e44" class="OCCURRENCE"> makes </EVENT>
    us role models. That was Eileen Collins, after she
```

Figure A.1: TimeML example (part 1). Taken from ABC19980304.1830.1636.tml in the Timebank corpus.

```
<EVENT eid="e10" class="OCCURRENCE"> flew </EVENT>
as the first ever co-pilot. Being commander is different. It means supervising the rest of the crew in training and leading them in flight. It is, in short,
the kind of management job many American women say they've had to fight for. In space, some say female pilots were
<EVENT eid="e20" class="OCCURRENCE"> held </EVENT>
up until
<TIMEX3 tid="t36" type="DATE" value="PRESENT_REF" temporalFunction="true" functionInDocument="NONE" anchorTimeID="t0"> now </TIMEX3>
by the
<EVENT eid="e51" class="STATE"> lack </EVENT>
of piloting opportunities for them in the military. Once Colonel Collins was
<EVENT eid="e22" class="I_ACTION"> picked </EVENT>
as a NASA
<EVENT eid="e52" class="STATE"> astronaut </EVENT>
, she
<EVENT eid="e23" class="OCCURRENCE"> followed </EVENT>
a normal progression within NASA. Nobody
<EVENT eid="e24" class="OCCURRENCE"> hurried </EVENT>
her up. No one
<EVENT eid="e25" class="OCCURRENCE"> held </EVENT>
her back. Many NASA watchers
<EVENT eid="e26" class="REPORTING"> say </EVENT>
female astronauts have
<EVENT eid="e27" class="OCCURRENCE"> become </EVENT>
part of the agency's routine. But they still have
<EVENT eid="e28" class="I_ACTION"> catching </EVENT>
up to do two hundred and thirty four Americans have
<EVENT eid="e30" class="OCCURRENCE"> flown </EVENT>
in space, only twenty six of them women. Ned Potter, ABC News.
</TEXT>
<MAKEINSTANCE eventID="e1" eiid="ei258" tense="PAST" aspect="NONE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e2" eiid="ei259" tense="PRESENT" aspect="PERFECTIVE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e3" eiid="ei260" tense="FUTURE" aspect="NONE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e37" eiid="ei261" tense="NONE" aspect="NONE" polarity="POS" pos="NOUN" />
<MAKEINSTANCE eventID="e38" eiid="ei262" tense="NONE" aspect="NONE" polarity="POS" pos="NOUN" />
<MAKEINSTANCE eventID="e40" eiid="ei263" tense="PRESENT" aspect="PERFECTIVE" polarity="POS" pos="NOUN" />
<MAKEINSTANCE eventID="e41" eiid="ei264" tense="NONE" aspect="NONE" polarity="POS" pos="NOUN" />
<MAKEINSTANCE eventID="e6" eiid="ei265" tense="PAST" aspect="NONE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e7" eiid="ei266" tense="PAST" aspect="NONE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e68" eiid="ei268" tense="PRESENT" aspect="PROGRESSIVE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e44" eiid="ei269" tense="PRESENT" aspect="NONE" polarity="POS" pos="VERB" />
```

Figure A.2: TimeML example (part 2). Taken from ABC19980304.1830.1636.tml in the Timebank corpus.

```
<MAKEINSTANCE eventID="e10" eiid="ei272" tense="PAST" aspect="NONE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e20" eiid="ei274" tense="PAST" aspect="NONE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e51" eiid="ei275" tense="NONE" aspect="NONE" polarity="POS" pos="NOUN" />
<MAKEINSTANCE eventID="e22" eiid="ei276" tense="PAST" aspect="NONE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e52" eiid="ei277" tense="NONE" aspect="NONE" polarity="POS" pos="NOUN" />
<MAKEINSTANCE eventID="e23" eiid="ei278" tense="PAST" aspect="NONE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e24" eiid="ei280" tense="PAST" aspect="NONE" polarity="NEG" pos="VERB" />
<MAKEINSTANCE eventID="e25" eiid="ei281" tense="PAST" aspect="NONE" polarity="NEG" pos="VERB" />
<MAKEINSTANCE eventID="e26" eiid="ei282" tense="PRESENT" aspect="NONE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e27" eiid="ei283" tense="PRESENT" aspect="PERFECTIVE" polarity="POS" pos="VERB" />
<MAKEINSTANCE eventID="e28" eiid="ei285" tense="NONE" aspect="NONE" polarity="POS" pos="NOUN" />
<MAKEINSTANCE eventID="e30" eiid="ei286" tense="PRESENT" aspect="PERFECTIVE" polarity="POS" cardinality="234" pos="VERB" />
<MAKEINSTANCE eventID="e30" eiid="ei288" tense="PRESENT" aspect="PERFECTIVE" polarity="POS" cardinality="26" pos="VERB" />
<MAKEINSTANCE eventID="e254" eiid="ei289" tense="PRESENT" aspect="PERFECTIVE" polarity="POS" pos="PREPOSITION" />
<TLINK lid="l31" relType="BEFORE" eventInstanceID="ei289" relatedToTime="t55" />
<TLINK lid="l1" relType="SIMULTANEOUS" eventInstanceID="ei261" relatedToEventInstance="ei262" />
<TLINK lid="l2" relType="IS_INCLUDED" eventInstanceID="ei262" relatedToTime="t33" />
<TLINK lid="l3" relType="BEFORE" eventInstanceID="ei263" relatedToTime="t0" />
<TLINK lid="l5" relType="IDENTITY" eventInstanceID="ei261" relatedToEventInstance="ei264" />
<TLINK lid="l6" relType="AFTER" eventInstanceID="ei266" relatedToEventInstance="ei265" />
<TLINK lid="l7" relType="IDENTITY" eventInstanceID="ei268" relatedToEventInstance="ei269" />
<TLINK lid="l11" relType="ENDED_BY" eventInstanceID="ei274" relatedToTime="t36" />
<TLINK lid="l12" relType="DURING" eventInstanceID="ei275" relatedToEventInstance="ei274" />
<TLINK lid="l13" relType="BEGINS" eventInstanceID="ei276" relatedToEventInstance="ei277" />
<TLINK lid="l14" relType="BEFORE" eventInstanceID="ei276" relatedToTime="t0" />
<TLINK lid="l15" relType="AFTER" eventInstanceID="ei278" relatedToEventInstance="ei276" />
<TLINK lid="l17" relType="IS_INCLUDED" eventInstanceID="ei258" relatedToTime="t32" />
<TLINK lid="l20" relType="INCLUDES" eventInstanceID="ei282" relatedToTime="t0" />
<TLINK lid="l21" relType="BEFORE" eventInstanceID="ei283" relatedToEventInstance="ei285" />
<TLINK lid="l22" relType="BEFORE" timeID="t0" relatedToEventInstance="ei286" />
<TLINK lid="l23" relType="INCLUDES" eventInstanceID="ei286" relatedToEventInstance="ei288" />
<TLINK lid="l24" relType="AFTER" eventInstanceID="ei260" relatedToTime="t0" />
<TLINK lid="l25" relType="BEGINS" eventInstanceID="ei265" relatedToTime="t34" />
<TLINK lid="l26" relType="ENDS" eventInstanceID="ei266" relatedToTime="t34" />
<SLINK lid="l27" relType="FACTIVE" eventInstanceID="ei260" subordinatedEventInstance="ei261" />
<SLINK lid="l28" relType="FACTIVE" eventInstanceID="ei258" subordinatedEventInstance="ei259" />
<SLINK lid="l29" relType="EVIDENTIAL" eventInstanceID="ei282" subordinatedEventInstance="ei283" />
</TimeML>
```

Figure A.3: TimeML example (part 3). Taken from ABC19980304.1830.1636.tml in the Timebank corpus.

B

Temporal Inference

This appendix shows the full set of relation inference results. The results are divided into 2 tables as below.

X \ Y	b	a	ib	ia	s	in	isin	d	di	beg	bb	e	eb
BEFORE (b)	b	undef	b	b, ib, d, beg, isin	b	b	b, ib, d, beg, isin	b, ib, d, beg, isin	b	b	b	b, ib, d, beg, isin	b
AFTER (a)	undef	a	a, ia, d, e, isin	a	a	a	a, ia, d, e, isin	a, ia, d, e, isin	a	a, ia, d, e, isin	a	a	a
IMMEDIATELY BEFORE (ib)	b	a, ia, di, eb, in	b	e, eb, s	ib	b	d, beg, isin	d, beg, isin	b	beg	beg	d, beg	b
IMMEDIATELY AFTER (ia)	b, ib, di, e	a	beg, bb, s	a	ia	a	d, e, isin	d, e, isin	a	d, e, isin	a	ia	ia
SIMULTANEOUS (s)	b	a	ib	ia	s	in	isin	d	di	beg	bb	e	eb
INCLUDES (in)	a, ib, di, eb, in	a, di, ia, bb, in	di, eb, in	di, bb, in	in	in, di	d, di, s, in, isin	d, di, s, in, isin	in, di	di, eb, in	di, in	di, bb, in	di, in
IS INCLUDED (isin)	b	a	b	a	isin	undef	isin, d	isin, d	undef	d, isin	a, ia, d, e, isin	d, isin	b, ib, d, beg, isin

Table B.1: Temporal relation inference. If A has relation X to B and B has relation Y to C, then A has relation Z to C. The inference relation Z is shown in the table.

X \ Y	b	a	ib	ia	s	in	isin	d	di	beg	bb	e	eb
DURING (d)	b	a	b	a	d	undef	d, isin	d, isin	undef	d, isin	a, ia, d, e, isin	d, isin	b, ib, d, beg, isin
DURING INV (di)	a, ib, di, eb, in	a, di, ia, bb, in	di, eb, in	di, bb, in	di	in, di	d, di, s, in, isin	d, di, s, in, isin	in, di	di, eb, in	di, in	di, bb, in	di, in
BEGINS (b)	b	a	b	ia	beg	b, ib, di, eb, in	d, isin	d, isin	b, ib, di, eb, in	beg	beg, bb, s	d, isin	b, ib
BEGUN BY (bb)	b, ib, di, eb, in	a	di, eb, in	eb	bb	di, in	d, e, isin	d, e, isin	di, in	beg, bb, s	bb	undef	di, in
ENDS (e)	b	a	ib	a	e	a, ia, di, eb, in	d, isin	d, isin	a, ia, di, eb, in	d, isin	a, ia	e	e, eb, s
ENDED BY (eb)	b	a, ia, di, bb, in	ib	bb, di, in	eb	di, in	d, beg, isin	d, beg, isin	di, in	undef	di, in	e, eb, s	eb

Table B.2: Temporal relation inference. If A has relation X to B and B has relation Y to C, then A has relation Z to C. The inference relation Z is shown in the table.

Bibliography

Allen, J. F. (1983), Maintaining knowledge about temporal intervals, *Commun. ACM*, *26*(11), 832–843, doi:10.1145/182.358434.

Bollegala, D., N. Okazaki, and M. Ishizuka (2006), A bottom-up approach to sentence ordering for multi-document summarization, in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pp. 385–392, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1220175.1220224.

Bos, J., and K. Markert (2005), Recognising textual entailment with logical inference, in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 628–635, Association for Computational Linguistics.

Chambers, N. (2013), Navytime: Event and time ordering from raw text, *Tech. rep.*, DTIC Document.

Chambers, N., and D. Jurafsky (2008), Jointly combining implicit constraints improves temporal ordering, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pp. 698–706, Association for Computational Linguistics, Stroudsburg, PA, USA.

Chambers, N., S. Wang, and D. Jurafsky (2007), Classifying temporal relations between events, in *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pp. 173–176.

Chang, A. X., and C. D. Manning (2012), Sutime: A library for recognizing and normalizing time expressions., in *LREC*, pp. 3735–3740.

Denis, P., and P. Muller (2011), Predicting globally-coherent temporal structures from texts via endpoint inference and graph decomposition, in *IJCAI-11-International Joint Conference on Artificial Intelligence*.

Do, Q. X., W. Lu, and D. Roth (2012), Joint inference for event timeline construction, in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 677–687, Association for Computational Linguistics.

Fan, R.-E., K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin (2008), Liblinear: A library for large linear classification, *J. Mach. Learn. Res.*, *9*, 1871–1874.

Fellbaum, C. (2010), Wordnet, in *Theory and applications of ontology: computer applications*, pp. 231–243, Springer.

Kolya, A. K., A. Kundu, R. Gupta, A. Ekbal, and S. Bandyopadhyay (2013), Ju_cse: A crf based approach to annotation of temporal expres-sion, event and temporal relations, *Atlanta, Georgia, USA*, p. 64.

Laokulrat, N., M. Miwa, Y. Tsuruoka, and T. Chikayama (2013), Uttime: Temporal relation classification using deep syntactic features, in *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pp. 88–92, Association for Computational Linguistics, Atlanta, Georgia, USA.

Laokulrat, N., M. Miwa, and Y. Tsuruoka (2014), Exploiting timegraphs in temporal relation classification, in *Textgraphs-9*, pp. 6–14.

Mani, I., M. Verhagen, B. Wellner, C. M. Lee, and J. Pustejovsky (2006), Machine learning of temporal relations, in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pp. 753–760, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1220175.1220270.

Manning, C. D., M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky (2014), The Stanford CoreNLP natural language processing toolkit, in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55–60.

Martins, A. F. T., D. Das, N. A. Smith, and E. P. Xing (2008), Stacking dependency parsers, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pp. 157–166, Association for Computational Linguistics, Stroudsburg, PA, USA.

Miller, S. A., and L. K. Schubert (1990), Time revisited, *Comput. Intell.*, *6*(2), 108–118, doi:10.1111/j.1467-8640.1990.tb00294.x.

Miyao, Y., and J. Tsujii (2008), Feature forest models for probabilistic hpsg parsing, *Comput. Linguist.*, *34*(1), 35–80.

Pustejovsky, J., R. Ingria, R. Saurí, J. Castaño, J. Littman, R. Gaizauskas, A. Setzer, G. Katz, and I. Mani (2005), The specification language timeml, *The Language of Time: A reader*, pp. 545–557.

Pustejovsky, J., et al. (2003), The TIMEBANK corpus, in *Proceedings of Corpus Linguistics 2003*, pp. 647–656, Lancaster.

Ravichandran, D., and E. Hovy (2002), Learning surface text patterns for a question answering system, in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pp. 41–47, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1073083.1073092.

Stenetorp, P., S. Pyysalo, G. Topić, T. Ohta, S. Ananiadou, and J. Tsujii (2012), Brat: A web-based tool for nlp-assisted text annotation, in *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pp. 102–107, Association for Computational Linguistics, Stroudsburg, PA, USA.

Tatu, M., and M. Srikanth (2008), Experiments with reasoning for temporal relations between events, in *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pp. 857–864, Association for Computational Linguistics, Stroudsburg, PA, USA.

UzZaman, N., and J. F. Allen (2010), Trips and trios system for tempeval-2: Extracting temporal information from text, in *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pp. 276–283, Association for Computational Linguistics, Stroudsburg, PA, USA.

UzZaman, N., and J. F. Allen (2011), Temporal evaluation, in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pp. 351–356.

UzZaman, N., H. Llorens, J. F. Allen, L. Derczynski, M. Verhagen, and J. Pustejovsky (2012), Tempeval-3: Evaluating events, time expressions, and temporal relations.

UzZaman, N., H. Llorens, L. Derczynski, M. Verhagen, J. Allen, and J. Pustejovsky (2013), Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations, in *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pp. 2–9, Association for Computational Linguistics, Atlanta, Georgia, USA.

Verhagen, M., R. Gaizauskas, F. Schilder, M. Hepple, G. Katz, and J. Pustejovsky (2007), Semeval-2007 task 15: Tempeval temporal relation identification, in *Proceedings of the 4th International Workshop on Semantic Evaluations*, pp. 75–80, Association for Computational Linguistics.

Verhagen, M., R. Saurí, T. Caselli, and J. Pustejovsky (2010), Semeval-2010 task 13: Tempeval-2, in *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pp. 57–62, Association for Computational Linguistics, Stroudsburg, PA, USA.

Wolpert, D. H. (1992), Stacked generalization, *Neural Networks*, *5*, 241–259.

Yoshikawa, K., S. Riedel, M. Asahara, and Y. Matsumoto (2009), Jointly identifying temporal relations with markov logic, in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, ACL '09, pp. 405–413, Association for Computational Linguistics, Stroudsburg, PA, USA.

PUBLICATIONS

Journals

1. Natsuda Laokulrat, Makoto Miwa, and Yoshimasa Tsuruoka. 2015. A Stacking Approach to Temporal Relation Classification with Temporal Inference. *Journal of Natural Language Processing.* (under review)

International Conferences

1. Natsuda Laokulrat, Makoto Miwa, and Yoshimasa Tsuruoka. 2014. Exploiting Timegraphs in Temporal Relation Classification. *Proceedings of TextGraphs-9: the workshop on Graph-based Methods for Natural Language Processing (EMNLP 2014)*, pp. 6–14.

2. Natsuda Laokulrat, Makoto Miwa, Yoshimasa Tsuruoka, and Takashi Chikayama. 2013. UTTime: Temporal Relation Classification using Deep Syntactic Features. *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pp. 88–92.

Domestic Conferences

1. Natsuda Laokulrat, Makoto Miwa, Yoshimasa Tsuruoka, and Takashi Chikayama. 2013. Enhancing Temporal Relation Classification by Features Extracted from Syntactic Parser. *NLP2013.*