Event Centric Approaches in Natural Language Processing

Yin Jou Huang

May 2021

Abstract

Humans use natural language to communicate with others about the events they observe. In the form of news articles, blog posts, or even tweets, an overwhelmingly large amount of texts are published every day. These natural language texts encode the various events that happen around us, presenting a rich repository of events from which we can mine invaluable knowledge about the rapidly changing world. However, the large amount of data has gone far beyond human ability to read and process manually. Thus, how to automatically extract and analyze the events in natural language texts has become an important task.

Due to their importance in natural language, events have been studied pervasively from many aspects. Some works analyze individual events and aim to identify their composing factors, such as event extraction, event coreference resolution, etc. Other works focus on the interactions and the various relations between multiple events, such as script learning (narrative event knowledge acquisition), discourse parsing, etc. In addition, many downstream tasks can benefit from incorporating events into the task, such as automatic summarization, question answering, etc.

In this thesis, we study the events in natural language texts. Specifically, we focus on three representative event-related tasks: event coreference resolution, narrative event relation knowledge acquisition, and automatic summarization. We first tackle the task of event coreference resolution, which is an important information extraction task. Event coreference resolution aims to identify and cluster the event mentions that refer to the same real-world event. We focus on the argument compatibility constraint of the task and propose a transfer learning framework of event coreference resolution and learn argument compatibility from

the abundant unannotated corpora available. We conducted experiments on the benchmark KBP corpora and verified the effectiveness of our proposed method.

Second, we tackle the task of narrative event relation knowledge acquisition. Narrative event relation is an important type of commonsense knowledge that captures the stereotypical ordering of temporally and causally related events. We adopt a 2-stage framework to extract narrative event relation knowledge from a large Japanese corpus. In the first stage, we utilize association rule mining algorithm to identify narrative-related events of statistic significance. In the second stage, we manually constructed an annotated corpus to learn the patterns of shared arguments between the event pairs. With the methods above, we collected a large amount of narrative event relation knowledge (400,000 pairs of narrative events) from the web corpus.

Last, we focus on the task of automatic summarization. We adopt the extractive summarization framework and extract salient elementary discourse units (EDUs) as summaries. We formulate the task as an event saliency identification problem and adopt a heterogeneous graph-based model for extractive summarization. We propose a heterogeneous document graph to model the interactions between text units of different granularity, including the coreference and discourse relations. A graph attention (GAT) based graph encoder is used to capture the structure of the heterogeneous document graph. We conducted experiments on the benchmark CNN/Daily Mail corpus and verified the effectiveness of our proposed method.

Acknowledgments

Throughout my PhD, I received a great deal of support and assistance from many people. Without them, the completion of this thesis would not have been possible.

First, I would like to express my deepest gratitude to my supervisor, Professor Sadao Kurohashi, for his invaluable supervision, guidance, and encouragement. I would like to thank him for providing his constant support during the six years of my study in the Kurohashi Laboratory of Kyoto University.

I am also grateful to the members of my thesis examination committee: Professor Tatsuya Kawahara and Professor Takayuki Ito, for their advice and feedback.

I would like to express my gratitude to Professor Vincent Ng of the University of Texas at Dallas, for the supervision of our joint work on event coreference resolution. The two-month internship in UT-Dallas is a valuable experience for me, I am very thankful for having the opportunity.

I would also like to thank Professor Daisuke Kawahara, Dr. Chenhui Chu, Dr. Yugo Murawaki, Dr. Fei Cheng, Dr. Ribeka Tanaka, Dr. Tomohide Shibata, and Dr. Fabien Cromieres for providing their expertise in guiding my research. I learned a lot from their invaluable suggestions and advice regarding my research through my PhD study.

I am blessed with many wonderful lab-mates of Kurohashi Laboratory. Especially, I would like to thank Dr. Raj Dabre, Dr. Tomohiro Sakaguchi, Dr. Shuhei Kurita, Mr. Tolmachev Arseny, Mr. Yudai Kishimoto, Mr. Tareq Alkhaldi, and Mr. Hirokazu Kiyomaru for having enlightening discussions with me on natural language processing research.

I also would like to thank Ms. Yuko Ashihara, Ms. Naho Yoshitoshi, and Ms. Terumi Kosugi, who are the secretaries of Kurohashi Laboratory, for all their

assistance in administrative matters.

Last but not least, I would like to thank my family and friends for their unconditional love and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Events form the backbone of our daily communication. Every day, tens of millions of events happen around the world. In order to communicate with others, humans use natural language to encode the information of the events they observed.

In the past decades, the increasing popularity of the Internet and the rise of the new media make it possible for everyone to express their thoughts effortlessly. The widespread of the Internet gives everyone free access to abundant and diverse sources of natural language texts, including news articles, blog posts, social media posts, just to name a few. These natural language texts are a rich repository of real-world events. By mining the events and studying the various interactions between them, we can obtain knowledge of the real world and develop applications that improve our everyday life in many aspects.

In this thesis, we study the events in natural language texts. An overwhelmingly large amount of new information is published through different media every day, which is far beyond human ability to read and process manually. Thus, how to automatically extract events of interest from text has been a major research problem in the information extraction (IE) and natural language processing (NLP) fields. In addition to identifying the occurrence of an event, we also want to know about the participants involved in the event and the different roles these participants play. Also, the same real-world event may be described multiple

times throughout a document or even across multiple documents in vastly different contexts and narrative styles. Thus, the identification of coreferent event mentions that describe the same real-world events is important for the purpose of aggregating information about an event.

Instead of describing only a single event, humans usually connect multiple events to form a complete story. Thus, recognizing how events interact with each other in texts is critical for natural language understanding. There exist various types of relations among events. For example, the temporal relation indicates the relative ordering of the occurring time of events, and the causal relation specifies the cause-effect relationship in which one event causes another event to occur. Related events are often ordered in a temporally and causally reasonable way in natural language text. The narrative event sequence is an important type of commonsense knowledge that captures this stereotypical ordering of events. Also, related events are connected through different rhetorical relations to form a coherent text. Building upon the various rhetorical relations between events, the discourse structure captures the high-level linguistic structure of a natural language text. The understanding of discourse structure is especially important for the NLP applications that operate on the scale of documents.

Due to the significant advance in deep learning and other machine learning techniques, the NLP field has witnessed a gradual shift of interest from entity-based tasks to event-based tasks in recent years. Some event-related tasks aim to identify the composing elements and the various aspects of a given event. These tasks are the major information extraction tasks, including event extraction, event argument identification, event coreference resolution, etc. Other event-related tasks focus on the interactions among multiple events. Tasks such as script (narrative event sequence) learning and discourse analysis are also important for modeling the global structure of events within natural language texts. The above event-related tasks can benefit NLP tasks such as automatic summarization, question answering, essay scoring, and many other tasks that require natural language understanding.

In this chapter, we will give a detailed introduction to the basics of the events. In Section 1.2, we first give the definition of event in natural language texts

and then introduce the two basic event-related tasks: event extraction and event coreference resolution. In Section 1.3, we study the interactions between events from the following two aspects: narrative event sequence (script) and discourse structure of documents. Finally, Section 1.4 includes the outline of the thesis.

1.2 Events in Natural Language Text

Events have always been one of the main focuses of research in the natural language processing (NLP) field. In Section 1.2.1, we first introduce the common definitions of events used in existing NLP researches. Then, we introduce the event extraction task in Section 1.2.2. The task of event extraction aims to detect the occurrences of events in natural language texts, and to identify the event type and the participating arguments of them. Last, we introduce the task of event coreference resolution in Section 1.2.3. The task of event coreference resolution aims to identify coreferent event mentions that describe the same real-world event, which is helpful in aggregating information about a given event.

1.2.1 Definitions of Event

The identification and analysis of events in natural language text have always been a topic of interest in NLP. Different researches may represent and define events differently depending on the specific purpose of the research topic. In the following, we introduce several formulations of events that are used in previous researches.

Syntactic Representation of Event

Some existing works adopted the syntactic representation of events by taking predicate-argument structure (PAS) as the basic unit of events. A PAS consists of a predicate, which is mostly a verbal expression, and its syntactic arguments, such as the subject, the direct object, the indirect object, etc. Figure 1.1 illustrates two events represented in PAS format.

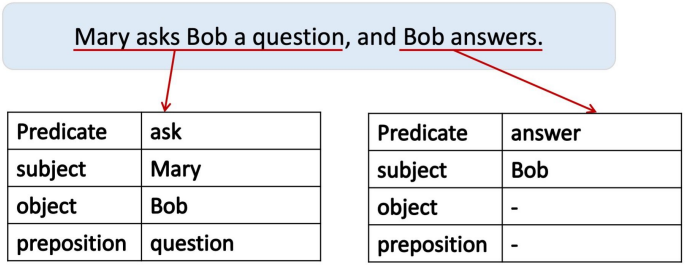


Figure 1.1: Syntactic representation of event: predicate-argument structure (PAS) as the basic unit of events.

In Japanese studies, people often incorporate case markers and their corresponding surface cases as the syntactic cases in predicate-argument structures (Figure 1.2). Main surface cases include ga-case(が格), wo-case(を格), and ni-case(に格), which roughly corresponds to the subjective (nominative), direct objective (accusative), and indirect objective (dative) cases.

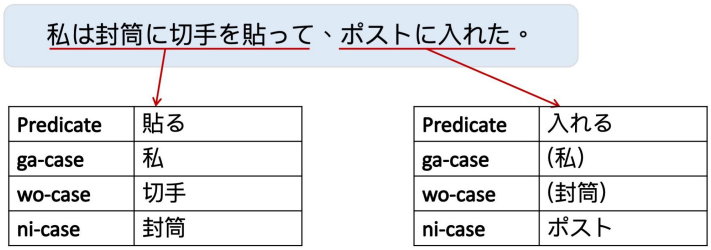


Figure 1.2: In Japanese studies, surface case markers are used to construct PAS.

By representing events with PAS, the syntactic representation of events does not require a specialized annotation corpus. Thus, adopting the syntactic representation of events enables us to extract and manipulate events on a large scale. For example, Chambers et al. [12] and Pichotta et al. [105] represent events as PAS and learn narrative event sequences automatically from a large-scale corpus.

Semantic Representation of Event

The semantic representations of events model the different participants of an event semantically. The definition of event introduced by the Automatic Content Extraction (ACE) [37] annotation scheme is the most representative of this category. In ACE formulations, an event consists of a trigger phrase and its corresponding arguments. The detailed definitions are summarized below:

- **Event mention**: the phrase or sentence within which an event is described.
- **Event trigger**: the word/phrase that most clearly expresses the occurrence of the event, typically verb or noun.
- **Event type**: the semantic class of the event.
- **Event argument**: the entity that serves as a participant or an attribute with a specific role in the event.
- **Argument role**: the relation between an argument and the event in which it participates.

Figure 1.3 shows an example of an event with the ELECT event type. The trigger is the word *elected*, and the explicit arguments are PERSON, POSITION, TIME, LOCATION arguments. ELECT events can also hold ENTITY arguments, but we keep the argument slot empty since the corresponding information is not present in the context.

Greg Lashutka was **elected** major of Columbus in 1993.

Argument role	Definition	Argument
PERSON	The person elected	Greg Lashutka
ENTITY	The voting agent(s)	-
POSITION	The job title for the position being nominated to	major of Columbus
TIME	When the election takes place	1993
LOCATION	Where the election takes place	Columbus

Figure 1.3: The example of an ELECT event.

Different event types have different semantic argument roles. For example, Figure 1.4 shows an ARREST-JAIL event with the trigger word *jailed*. Different from the ELECT events, ARREST-JAILS events can take PERSON, AGENT, CRIME, TIME, and PLACE arguments.

Abu Talb, the last mayor prosecution witness, has been **jailed** in Sweden for attacks against Jewish and American targets in Europe.

Argument role	Definition	Argument
PERSON	The person who is jailed of arrested	Abu Talb, the last mayor prosecution witness
AGENT	The jailer or the arresting agent(s)	-
CRIME	The crime for which the PERSON-arg is being jailed of arrested	attacks against Jewish and American targets in Europe
TIME	When the person is jailed or arrested	-
PLACE	Where the person is jailed or arrested	-

Figure 1.4: The example of an ARREST-JAIL event.

The semantic representation of events is very popular among researchers that handle events of specific types of interest, such as closed-domain event extraction, event coreference resolution, etc. Due to the popularity, manual efforts have been put into the construction of annotated event corpus.

Elementary Discourse Unit

Discourse analysis studies how events are combined to form natural and coherent texts. Being one of the most popular theories of discourse analysis, the **Rhetoric Structure Theory (RST)** framework segments the article into Elementary Discourse Units (EDU), which is the minimal units of discourse (Figure 1.5). As can be observed from Figure 1.5, the EDUs are typically clauses (consisting of a predicate and its arguments) and can be considered as the pieces of events composing the document.

Different from the syntactic and semantic representations of events, the EDUs do not have explicit argument structures. The unstructured nature and natural

language-like characteristic of EDUs provide the following advantages: (1) free from error propagation of argument extraction components, and (2) straightforward for human eyes to examine. Representing events in the natural language-like form such as EDUs is popular among research works that focus on the relations and interactions between events.

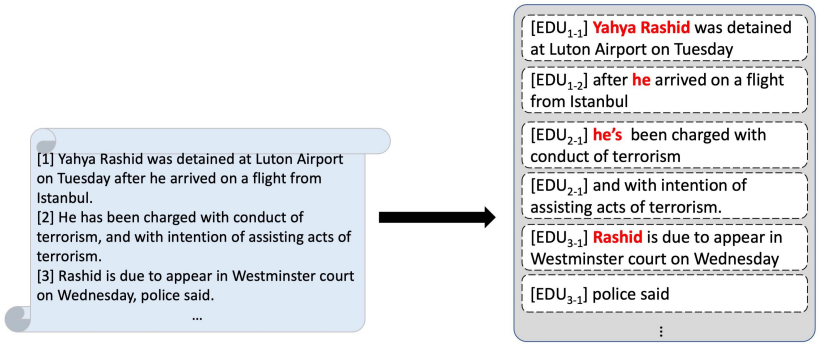


Figure 1.5: Trigger and semantic arguments as an unit of event.

1.2.2 Event Extraction

Event extraction is an important yet challenging task that serves as a crucial first step for various downstream NLP tasks, such as event coreference resolution, event relation extraction, and any task that handles events.

The goal of event extraction is to detect occurrences of events in natural language texts. Further, the detected events are classified into different event types and the participating arguments and associated attributes are also identified. The key idea of event extraction is to extract structured representations of events from unstructured natural language texts. The structured representation of an event should directly or indirectly help in answering the "5W1H" questions - who, when, where, what, why, and how - about the given event.

The task of event extraction can be further classified into closed-domain and open-domain event extraction, which differ in the existence of predefined event types and require different extraction strategies. In the following, we introduce the task formulations, annotated corpora, and the popular approaches of event extraction.

Closed-Domain Event Extraction

Closed-domain event extraction assumes the existence of predefined event types. For each predefined event type, an event schema specifying the possible argument roles is defined. Based on the predefined event schema, events of particular types are extracted from text. Also, the arguments of the event need to be identified and fill into the corresponding argument roles of the predefined event schema. For example, Figure 1.3 and Figure 1.4 represent the event schema of ELECT and ARREST-JAIL event types, respectively.

Closed-domain event extraction only focuses on specific types of events, and would neglect events that are not of the predefined event types. The set of predefined event types and their event schema varies across different research programs and domains.

Open-Domain Event Extraction

Open-domain event extraction does not assume any predefined event structures. The goal of open-domain event extraction is to identify previously undefined events. Also, in most cases, similar events are further clustered based on the keywords of the extracted events. Here, keywords refer to the words or phrases that describe an event, which can be the trigger or arguments of the event.

Most works of this line focus on the detection and clustering of events, and the identification of the arguments and argument roles are often omitted. However, some works also work on the induction of event schemas by assigning the event type label and attribute labels to each event cluster.

Open-domain event extraction is very useful for downstream applications that require board-coverage, dynamically-evolving, or domain-specific event categories. In addition to standard news articles, online social media posts are also popular targets for open-domain event extraction.

Event Extraction Corpora

Information extraction from unstructured natural language text has attracted a lot of interest among researchers in NLP. To encourage the research of information

extraction, many public evaluation programs have proposed their own task definitions and annotation schemes as well as publicly available annotated corpora to attract research interest. In the following, we introduce four important corpora commonly used in event extraction research:

1. **MUC corpus**:
   The Message Understanding Conference (MUC) program [122, 123] is the first public evaluation program for information extraction. Organized by the Defense Advanced Research Projects Agency (DARPA), the MUC conference was held seven times from 1987 to 1997. The MUC program introduced the slot filling task that aims to identify the structured information from natural language text, as can be seen as the first corpus for event extraction.

2. **ACE corpora**:
   The Automatic Content Extraction (ACE) program [37] is so far the most influential public evaluation program in this field, and most existing works of closed-domain event extraction follow the formulations of the ACE program.

   The ACE program covers various information extraction tasks, including the extraction of entities, time expressions, values, events as well as relations. For event extraction, the ACE 2005 benchmark corpus targets 8 event types, which can be further divided into 33 event subtypes (Table 1.1). For each event subtype, a specific set of argument roles are defined (like the ones in Figure 1.3 and Figure 1.4). The ACE 2005 benchmark corpus contains 599 documents and about 6000 manually labeled events.

3. **TDT Corpora**:
   The Topic Detection and Tracking (TDT) program [4] promotes open-domain event extraction. The main target of the TDT program is the identification and tracking of news events in series of news articles. The TDT corpora (TDT-1 to TDT-5) contain millions of news stories annotated with hundreds of topics (events) collected from multiple sources.

4. **TAC-KBP Corpora**:
   The event track of the Text Analysis Conference Knowledge Base Population

(TAC-KBP) program [40, 41, 44] contains several event-related tasks. The TAC-KBP program not only covers the closed-domain event extraction task (event nugget detection and event argument extraction), but also covers the tasks of event and entity coreference resolution (Event coreference and event argument linking).

Event Type	Event Subtypes
Life	Be-Born, Marry, Divorce, Injure, Die
Movement	Transport
Conflict	Attack Demonstrate
Business	Merge-Org, Declare-Bankruptcy, Start-Org, End-Org
Transaction	Transfer-Money, Transfer-Ownership
Personnel	Elect, Start-Position, End-Position, Nominate
Justice	Arrest-Jail, Execute, Pardon, Release-Parole, Fine, Convict, Charge-Indict, Trail-Hearing, Acquite, Sentence, Sue, Extradite, Appeal

Table 1.1: Predefined event (sub)types in ACE corpus.

Approaches

As one of the most important research subjects in NLP, event extraction has been intensively studied in the past decades. In the following, we introduce major existing approaches of event extraction:

Pattern Matching Pattern matching techniques are the earliest approaches of event extraction. Event templates are constructed manually, and the input texts which match such patterns are identified [112, 57]. Since the design of patterns is very labor-intensive and suffers from low coverage problem, some research works use weak supervision or bootstrapping to acquire patterns automatically by only relying on a small set of seed patterns [113, 132].

Supervised Learning Approaches Pattern matching techniques enjoy high extraction accuracy by using high-quality event patterns manually designed by domain experts. However, constructing a large number of event patterns of high quality is difficult. Thus, the research focus of event extraction has shifted to supervised machine learning based approaches that learn classifiers from annotated data.

Event extraction can be divided into two subtasks: trigger extraction and argument extraction. Here, trigger extraction includes both trigger identification and event type classification, and argument extraction includes both argument identification and argument role classification. The two subtasks can be executed in a pipelined manner, in which the classifiers for each subtask are trained independently [2, 53]. Since the pipeline execution suffers from error propagation problem, joint classification approaches are also proposed [68, 14].

Traditional machine learning algorithms such as support vector machine (SVM) or maximum entropy learning are based on manually designed linguistic features. Commonly used features include lexical features such as word lemma and part-of-speech (PoS) tag, syntactic features based on dependency parse tree, and semantic features such as synonyms and event/entity types.

In recent years, deep learning based approaches have merged to remove the need for feature engineering and increase model adaptability [96, 95, 70]. A deep learning model consists of multiple layers of artificial neurons, which convert the input features gradually into more and more abstract representations layer by layer. Event extraction models based on various deep learning architecture has been proposed, including convolutional neural networks (CNN), recurrent neural networks (RNN), and graph neural networks (GCN), etc. Since the training of deep learning models is very data-consuming, many works have explored semi-supervised learning techniques and use automatically generated training data to alleviate data deficiency problems.

Unsupervised Learning Approaches The unsupervised learning approaches are commonly used for open-domain event extraction. Open-domain event extraction mainly focuses on detecting event keywords and cluster sentences or articles

expressing similar events. Based on the keywords of an event, different clustering algorithms are used to cluster similar events into latent event groups [13, 114].

1.2.3 Event Coreference Relation

An event may be mentioned multiple times within a document or even across multiple documents. The coreference relation holds between two event mentions that describes the same real-world event. Consider the following example:

> *The ruling KMT Party in Taiwan is scheduled to* $\boldsymbol{elect}_{e_1}$ *a new chairperson in January 2012, it was decided at the top-level KMT* $\boldsymbol{meeting}_{e_2}$ *on Wednesday. KMT chairman Ma Ying-jeou stressed fair play during the* $\boldsymbol{election}_{e_3}$ *at Wednesday's* $\boldsymbol{meeting}_{e_4}$*.*

In the example above, we can see that the event mentions $\boldsymbol{elect}_{e_1}$ and $\boldsymbol{election}_{e_3}$ refer to the same real-world ELECT event. In this case, we say that the event mentions e_1 and e_3 are coreferent. Similarly, the event mentions e_2 and e_4 are also coreferent, since they both refer to the same real-world MEET event.

The task of event coreference resolution aims to identify the coreference relations between event mentions and cluster them into groups of distinct real-world events. Event coreference resolution is one of the main tasks of information extraction. The task can be further categorized into within-document and cross-document event coreference resolution. The majority of the existing works focus on **within-document event coreference**, which takes a single document as input and identifies the coreference relations between event mentions in the document. Within a given document, a real-world event might be mentioned multiple times from different perspectives. By identifying event mentions that correspond to the same real-world event, we can collect the complete information related to the specific event. On the other hand, there has been an increasing interest in the task of **cross-document event coreference resolution**, which aims to identify event coreference relations both within the same document and across different documents. Since important events tend to be reported in multiple different media sources with different phrasings and narrative styles, the identification of coreference relations across different documents is a challenging yet important

problem.

While seemingly similar, the task of event coreference resolution is quite different from the task of entity conference resolution. First, the diversity of event mentions is larger than that of entity mentions. While the entity mentions are mostly noun phrases, event mentions include more diverse syntactic objects, such as noun phrases, verb phrases, clauses, and sentences. Second, the distribution of coreferential event mentions is more sparse than coreferential entity mentions. The main reason for this is that the linguistic motive behind event coreference is different from that of entity coreference. In an article, the main entities (protagonists) are involved in many events and are referred to every time such events are described. On the other hand, an event is mentioned multiple times only when new aspects or information of the given event are introduced. Compared to entity coreference, event coreference is a more complex phenomenon that involves discourse motive to organize and consolidate the related contents of an event.

In the following, we introduce existing annotated corpora and popular approaches of event coreference resolution.

Event Coreference Corpora

1. **ACE corpora**:
   The Automatic Content Extraction (ACE) program [37] provides annotations of within-document event coreference resolution, which focus only on specific events types (as in Table 1.1). The most widely used ACE 2005 corpus contains around 600 English documents and 500 Chinese documents.

2. **OntoNote corpus**:
   The OntoNote corpus [108] consists of 600 documents based on news articles of Wall Street Journal and covers annotations of both entity and event coreference relations. While being a standard benchmark corpus for entity coreference resolution, the OntoNote corpus is not as popular for the task of event coreference resolution due to its coverage issues. Also, the OntoNote corpus covers both within-document and cross-document event coreference relations, but only for events with nominal event mentions.

3. **ECB corpus**:
   The EventCorefBank (ECB) corpus [8] mainly focuses on the task of cross-document event coreference resolution. Consisting of 1000 documents based on Google news, the corpus is annotated with inter-document coreference relations of 47 predefined event types. While ECB corpus also provides annotations of within-document event coreference relations, these intra-document links are only partially annotated. The ECB+ corpus [33] extends the original ECB corpus with a revised annotation scheme and a larger number of documents.

4. **TAC-KBP corpora**:
   The TAC-KBP program [40, 41, 44] focus on within-document event coreference task targeting 9 events types and 38 subtypes. The definitions of tasks and terminologies follow that of the ACE program. Being the largest existing corpus for within-document event coreference resolution, the TAC-KBP corpora contain 1000 English, 800 Chinese, and 400 Spanish annotated documents.

Approaches

In the following, we summarize the popular approaches and inference strategies that have been proposed for event coreference resolution over the past decades:

Mention Pair Model The most simple strategy for event coreference resolution is to adopt a two-stage framework. In the first stage, a binary classifier is used to calculate the likelihood of coreference between a pair of event mentions. Classifiers such as SVM [16], decision trees [34], maximum entropy [22], and neural networks [94] has been used. In the second stage, a separate clustering algorithm is used to coordinate the pairwise decisions made in the first stage. Popular clustering algorithms include agglomerative clustering, closest-first clustering, best-first clustering, etc.

Mention Ranking Model The pairwise decisions of mention pair models are independent of other candidate antecedents. On the other hand, the mention

ranking models consider the relative suitability of all candidate antecedents of an event mention by ranking them. Work such as Lu et al. [78] belongs to this category.

Easy-First Model Among all the coreference links in the document(s), some of them are easier to resolve than others. The idea of the easy-first model is to make decisions for the easier coreference links first, and utilize the results to help narrowing down the choices of the more difficult coreference decisions. Works like Liu et al. [72], Lu et al. [75], and Choubey et al. [26] belong to this category.

Joint Model Many event coreference models rely on the upstream event extraction module in a pipeline fashion. In order to alleviate the error propagation problem, several works have proposed systems that perform joint inference of event coreference and upstream tasks. Works based on integer linear programming (ILP) and Markov logic networks (MLN) have been proposed ([19], [80]).

1.3 Interactions between Events

Natural language can not only describe single events. Instead, humans usually organize multiple events together to form a complete story. This naturally acquired ability to construct and perceive such an organized sequence of events can be explained by the theories of narrative and discourse, along with the various relations between events. For the purpose of natural language understanding, it is important to understand how events interact with each other in texts. In this section, we will cover the narrative and discourse theories of natural language and introduce important relations between events, such as temporal relation, causal relation, discourse relation, etc.

1.3.1 Narrative Event Sequence

Human minds have the tendency to explain the events happening around us. Driven by such an explanatory nature, we constantly and often subconsciously mine the patterns in the past event sequences, make observations of the present

events, and attempt to predict future events. From our experience, we develop understanding of the world revolving around us and acquire the typical scenarios which occurred repeatedly. For example, if we observe the event 'John goes into a restaurant', we are likely to expect events such as 'John takes the menu' or 'John orders food' to happen subsequently.

Narrative event sequences are used to model such a common scenario. A narrative event sequence is a stereotypical sequence of events ordered in a temporally and causally reasonable way. Narrative event sequences reflect the commonsense knowledge of how we perceive and understand the world, or to be more specific, what kinds of scenarios do we consider normal. Also, the concept of narrative event sequences has a great influence on how we communicate. Based on the presumption of the commonsense knowledge of typical event sequences, it is possible for our communication to be carried out smoothly and efficiently.

The research of narrative event sequence has a long history in NLP. In the 1970s, Schank et al. [117] first proposed the concept of **scripts**, which represent the knowledge about the stereotypical event sequences under certain scenarios. For example, Figure 1.6 shows a restaurant script that models the sequence of events that about a common scenario that a customer enters a restaurant, orders food, eat the food, and finally pay the bill and exit the restaurant. The script involves four human participants, the *CUSTOMER*, the *WAITER*, the *COOK*, and the *CASHIER*, along with other object arguments. Not only the ordering of events, but the interactions of the participants (arguments) also serve as important commonsense knowledge that is beneficial to other NLP tasks such as entity coreference resolution, etc.

The analysis and modeling of narrative event sequences are beneficial for various applications that require natural language understanding, such as question answering, summarization, dialogue generation, next event prediction, etc. In the following, we introduce the various aspects of narrative event sequences, including event co-occurrence, temporal relation, causal relation, and sub-event structure. Also, we summarize the existing works on narrative event sequence acquisition (script learning) in NLP.

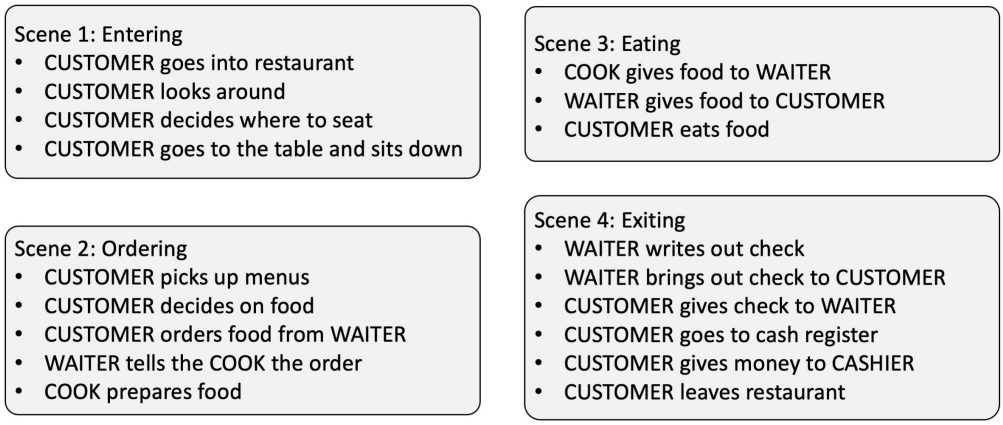


Figure 1.6: The restaurant script. Adapted from Schank et al. [117].

Event Co-occurrence

Some combinations of events appear together more often than would be expected by chance, this phenomenon is called event collocation or event co-occurrence in corpus linguistics. The textual co-occurrence of events is a measure of the relatedness of the events. Thus, capturing the co-occurrence patterns of events in text corpora is the most straightforward way to mine the narrative relations between events.

Temporal Relation Component

Temporal relation defines the ordering of events with respect to their occurring time. The topic of temporal reasoning has always been the main focus of many NLP studies. The ability to reason about how events unfold in time is a core aspect of how humans structure their knowledge about the world.

Typically when we talk about events, we often associate them with various temporal cues. Natural language text contains an abundant amount of temporal cues about the timing of the events. These temporal cues give information about the start time, end time, the duration time of the event, and can either be specified in absolute time or relative to other events or time expressions. The goal of temporal reasoning is to identify the temporal cues in text and build a complete overview of the temporal relations between the events within the given text.

The benchmark TimeBank corpus [109] (and the corpora following this line) provides an annotation scheme to annotate the time expressions as well as the temporal relations between events. While more fine-grained categorization is also possible, the coarse categories of temporal relations include the following: before, after, includes, included, simultaneous, vague.

Temporal relation depicts the ordering of how events unfold in time and serves as an important aspect of event narrative relation.

Causal Relation Component

Causal relation describes the cause-effect relation between two events where one event (*cause event*) is responsible for the occurrence of the other event (*effect event*).

In some cases, causality is expressed via explicit causality markers. The causality markers include causal connectives (*because, with the results that, since, ...*), causal verbs (*cause, bring about, ...*), etc. The presence of the causality marker is a direct clue to the presence of the causality relation between events. However, in some cases of causally related events, the causality markers do not exist. Without the presence of the causal markers, the identification of the causal relations is more challenging. Examples of explicit and implicit causal relations follows:

(a) *[I went to the restaurant]$_{effect}$* ***because*** *[one of my collegues recommended it.]$_{cause}$.*

(b) *[Be careful.]$_{effect}$ [The floor is wet.]$_{cause}$*

The benchmark Causal TimeBank corpus [91] introduces three categories of causality: Cause, Enable, and Prevent. Their representative verbal causality markers are summarized below:

1. CAUSE: cause, prompt, force, ...
2. ENABLE: enable, allow, help, ...
3. PREVENT: prevent, block, restrain, ...

Causality is a pervasive phenomenon in natural language text and an important component of event narrative relation. Causal relations play an important role of connecting events in a logical and meaningful way.

Sub-Event Relation

In natural language texts, event mentions describe events of different granularity. An event of larger granularity may contain sub-events of smaller granularity. A sub-event is an event that happens as part of its parent event both spatially and temporally. For example, a parent event *summit* may include sub-events such as *conversation* or *encounter*; while a parent event *attack* may include sub-events such as *capture*, *killing*, or *wounding*.

The knowledge of the internal structure of an event facilitates the reasoning over events. The sub-event relations account for the hierarchy nature of the events. Consider the following example:

> *Obama **sparred** with Vladimir Putin over how to end the **war** in Syria on Monday during an icy **encounter** at a G8 **summit**. **Speaking** after **talks** with Obama, Putin said they **agreed** the **bloodshed** must end.*

In the example above, the hierarchical structure of events characterized by the sub-event relations is:

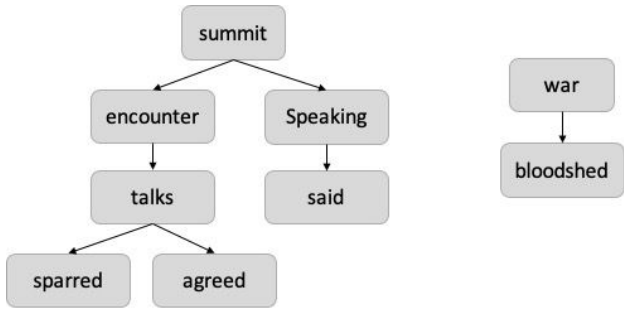


Figure 1.7: The hierarchical sub-event structure. Adapted from Glavaš et al. [45].

Unlike the temporal and causal relations which focused on the horizontal inference of events, the sub-event relation aims to capture the vertical relations

between events. Previous studies suggest that the sub-event structure is not only crucial for event understanding but also helpful for the identification of the horizontal relations between events.

Script Learning

Scripts encode the narrative event sequences, and the task of automatic script learning (script learning) has always been a topic of interest in NLP. In the following, we summarize the existing works on script learning.

The studies on script knowledge date back to the 1970s [117]. Instead of focusing on the automatic induction of script, the earliest works use manually generated scripts on tasks such as question answering (QA) or inference. Since the manually generated scripts are labor-intensive, they are limited in coverage and do not generalize well.

Later works proposed statistical methods to learn script automatically, most utilizing the co-occurrence statistics between events. Chambers et al. [12] used Pointwise Mutual Information (PMI) as the measure of co-occurrence between events and extract scripts (named **narrative chains** in this work) centering around a specific actor (the protagonist). Other works like Jans et al. [52], Pichotta et al. [105], Rudinger et al. [115] followed a similar paradigm.

With the popularity of neural network based methods in NLP, some works used recurrent neural networks to model the event sequence in text and make predictions about the next event [107, 106]. Granroth et al. [47] first introduced embeddings to represent events and achieved good results on the task. After that, embedding-based methods have been a dominant approach in this area. Other neural network based methods learned event representations specialized for the task of script learning [130, 63].

Some works explicitly incorporated external knowledge and considered event relations beyond event co-occurrence. Lee et al. [64] considered the discourse relations between events. Ding et al. [36] incorporated sentiment and intention information of events. Lv et al. [84] directly integrated temporal and causal event relations into script learning.

1.3.2 Discourse Structure of Natural Language Documents

While narrative event sequences model the stereotypical ordering of events that reflects standard scenarios, theory of discourse structure adopt a different point of view and focus on the various rhetorical relations in a document. Suppose that you have collected a handful of story-worthy events, how do you connect them into coherent text? This is where the discourse theory comes to play. The discourse theory aims to uncover the discourse structure of natural language texts and answer the question of how smaller text units are combined to form larger coherent texts such as paragraphs and articles.

The discourse structure of a document describes the high-level linguistic structure of a natural language text. Thus, identifying and incorporating discourse information is indispensable for achieving document-level language understanding. Document-level machine translation, summarization, sentiment analysis, and essay scoring are among the many downstream tasks that can benefit from incorporating information about discourse structure. In the following, we first introduce the theory of discourse structure and briefly discuss event saliency, which is a related concept of discourse theory.

Discourse Structure Theory

The **Rhetoric Structure Theory (RST)** [85] is the most influential theory of discourse analysis. Although other theories of document-level discourse structure have also been proposed, the majority of research works on document-level discourse analysis are based on the formulations of RST.

RST models the discourse structure of each document with a continuous constituency tree (also called RST tree, Figure 1.8). The leaf nodes of the RST tree represent the Elementary Discourse Units (EDU), which are the minimal units of discourse. EDUs are typically clauses (consisting of a predicate and its arguments), which can be considered as the units of events comprising the document. The adjacent EDUs are connected to each other through different rhetorical relations to form larger discourse units. By recursively connecting adjacent discourse units, we can eventually obtain the RST discourse tree of the document. For

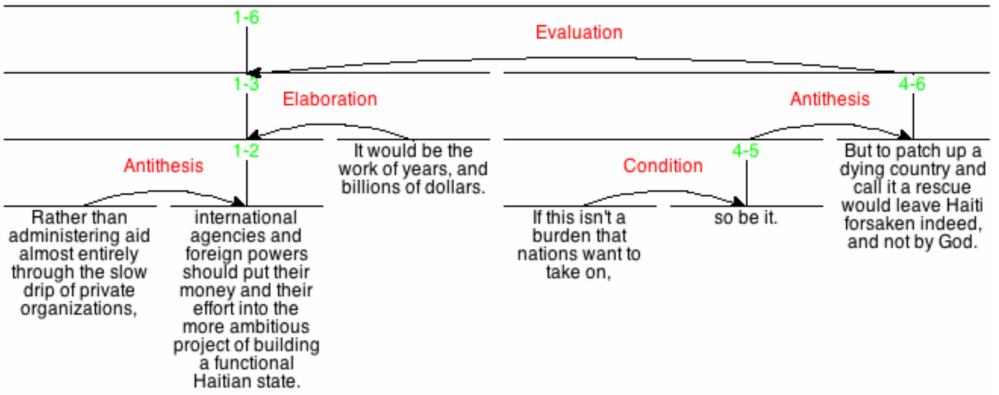


Figure 1.8: The constituency tree of RST. From Stede et al. [121] (Source: George Packer: Suffering. *The New Yorker 42, 2010*)

example, the text in Figure 1.8 consists of 6 EDUs. These EDUs are gradually combined into larger discourse units *1-2*, *4-5*, *1-3*, *4-6* and finally into the root node covering the full range of text (*1-6*).

RST introduces various rhetorical relations (Table 1.2). Each rhetorical relation describes a distinct rhetorical function by which two adjacent discourse units are connected. In addition, each discourse unit is categorized as either *NUCLEUS* or *SATELLITE* to indicate their relative saliency (importance). Generally speaking, *NUCLEUS* units describe more important information while *SATELLITE* units contain supplementary information. The rhetorical relations can be divided into two categories based on the relative saliency of the discourse units the given relation connects. For the first category, **mononuclear relations** capture the subordinate relation between a *NUCLEUS* unit of greater importance and a *SATELLITE* unit of less importance. For the second category, **multinuclear relations** capture the coordinate relation between two or more *NUCLEUS* units of similar importance.

Most of the discourse relations in RST are mononuclear relations, while multinuclear relations include *Sequence*, *Contrast*, and *List*, and *Joint* relations. In Figure 1.8, we can see that the mononuclear relations such as *Elaboration*, *Background*, and *Preparation* are illustrated by a directed arrow going from the *SATEL-*

Relation Type	Nucleus	Satellite
Antithesis	ideas favored by the author	ideas disfavored by the author
Background	text whose understanding is being facilitated	text for facilitating understanding
Circumstance	text expressing the events or ideas occurring in the interpretive context	an interpretive context of situation or time
Concession	situation affirmed by author	situation which is apparently inconsistent but also affirmed by author
Condition	action or situation whose occurrence results from the occurrence of the conditioning situation	conditioning situation
Elaboration	basic information	additional information
Enablement	an action	information intended to aid the reader in performing an action
Evaluation	a situation	an evaluative comment about the situation
Evidence	a claim	information intended to increase the reader' s belief in the claim
Interpretation	a situation	an interpretation of the situation
Justify	text	information supporting the writer' s right to express the text
Motivation	an action	information intended to increase the reader' s desire to perform the action
Cause	a situation	another situation which causes that one
Result	a situation	another situation which is caused by that one
Otherwise	action or situation whose occurrence results from the lack of occurrence of the conditioning situation	conditioning situation
Purpose	an intended situation	the intent behind the situation
Restatement	a situation	a re-expression of the situation
Solutionhood	a situation or method supporting full or partial satisfaction of the need	a question, request, problem, or other expressed need
Summary	text	a short summary of that text
Contrast	one alternate	-
Sequence	an item	-
List	an item	-
Joint	(unconstrained)	-

Table 1.2: Rhetorical relations in RST.

LITE node to the *NUCLEUS* node. On the other hand, the multinuclear *Contract* relation is represented only with undirected edges in the RST tree.

Table 1.2 lists the rhetorical relations in RST [85] along with the descriptions of their components. Based on the above set of rhetorical relations, more fine-grained or coarse-grained categorizations have also been proposed.

Event Saliency and Main Events

Within a given text, some events are more salient than others, they provide key information that is important for the progress of the story, and play a central role in the discourse structure. The research on event saliency aims to measure

the importance of events within a document, and to distinguish the salient events that are most relevant to the main content of the document.

Event saliency is an important aspect of discourse analysis. RST uses the label of *NUCLEUS* and *SATELLITE* to capture the relative saliency of rhetorically related discourse units. In addition to the localized setting above, the concept of event saliency can also be applied to a more global setting. The following are two **saliency tests** that have been considered in existing works on event saliency:

1. Summarization test:
   The summarization test of saliency is originally used in the entity saliency research [39], buy has also been applied to event saliency research. According to the definition, the events that are likely to be included in human written summaries are considered to be salient events. Liu et al. [74] utilized the summarization test of event saliency and built a large-scale event saliency corpus.

2. Coherence test:
   The coherence test of event salience considers an event to be salient if removing it greatly harms the coherence of the text. Otake et al. [99] utilized the coherence test of event saliency and proposed an unsupervised event saliency detection model.

The salient events are also called **main events** in some literature. The main events play an important role in the discourse structure of the article. For example, according to Van Dijk's theory of news discourse [125], the discourse structure of a news article revolves around the main events. In Van Dijk's theory, a news article consists of one or more main events and relevant events that are related to the main events in different manners, such as the background events that cause the main event, similar historical events that happened in the past, etc. Figure 1.9 shows the complete hierarchical structure of discourse functions proposed by [125].

The identification of main events is important for the analysis of various event relations. Previous research has reported on the heavy involvement of main events

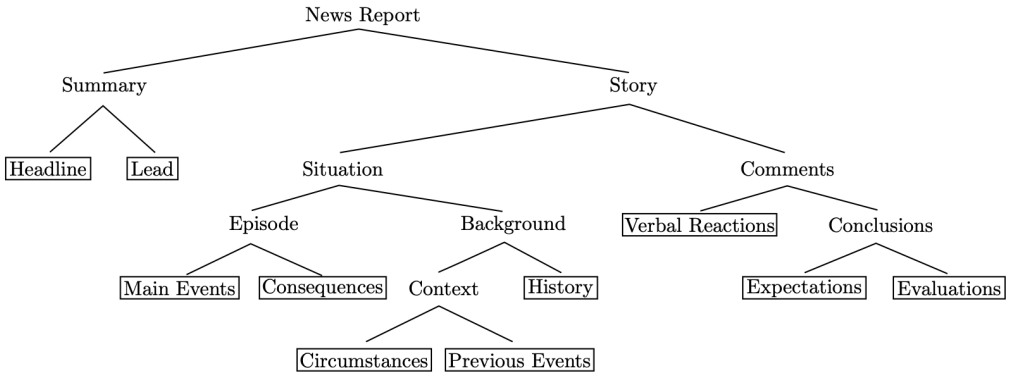


Figure 1.9: Van Dijk's theory of news discourse structure. From Yarlott et al. [137]

in causal relations within a document [43]. Also, main events are especially critical for the task of event coreference resolution [28]. Most of the (non-salient) events and their coreference links only appear locally within the article. On the other hand, the main events appear throughout the entire article and serve the important discourse function of content organization.

1.4 Outline of the Thesis

Due to their importance in natural language, events have been studied pervasively in previous research. The numerous event-related tasks have attracted a lot of research interest in the NLP field. Some works focus on the analysis of a single event, such as event extraction and event coreference resolution. Other works study the interactions and the various relations between events, such as script learning (narrative event knowledge acquisition), temporal/causal relation identification, discourse parsing, etc. In addition, downstream tasks such as automatic summarization, question answering (QA), sentiment analysis can also benefit from incorporating events into the main tasks.

In this thesis, we present three event-related tasks in NLP: event coreference resolution, narrative event relation knowledge acquisition, and automatic summarization. In Chapter 2, we introduce our work on event coreference resolution,

which is an important task that identifies event mentions that correspond to the same real-world event. We focus on the argument compatibility aspect of event coreference resolution task, and propose a transfer learning framework to learn argument compatibility from large unannotated corpus. In Chapter 3, we introduce our work on narrative event relation knowledge acquisition from large Japanese corpus. Japanese is a language of abundant omitted arguments. Thus, we focus on the improvement of shared argument identification of narrative-related event pairs. In Chapter 4, we introduce our work on automatic summarization. In this work, we perform extractive summarization on the EDU level. By viewing the EDUs as events, we can reframe the extractive summarization task as an event saliency identification problem. We propose a heterogeneous graph based method that captures the discourse relations and coreference relations between events and entities simultaneously. Finally, we present the conclusion of this thesis in Chapter 5.

Chapter 2

Transfer Learning Based Event Coreference Resolution

In this chapter, we focus on the task of event coreference resolution. Event coreference resolution is an important task of information extraction. By identifying text mentions that refer to the same real-world event, we can consolidate the information about a specific event.

In this work, we analyze the components of events and the two necessary conditions of coreferential event mentions: trigger relatedness and argument compatibility. We focus on the argument compatibility condition, which is more difficult to learn from the existing moderate-size labeled data. We propose a transfer learning based framework for event coreference resolution and learn argument compatibility from a large unlabeled corpus. Experiments on the KBP benchmark dataset confirm the effectiveness of our proposed method.

2.1 Introduction

As introduced in Section 1.2.3, event coreference resolution is a fundamental NLP task that aims to identify event mentions that correspond to the same real-world event. Here, we focus on within-document event coreference resolution and group event mentions within a document into coreferent clusters. Figure 2.1 shows a document consisting of three events described by six different event mentions.

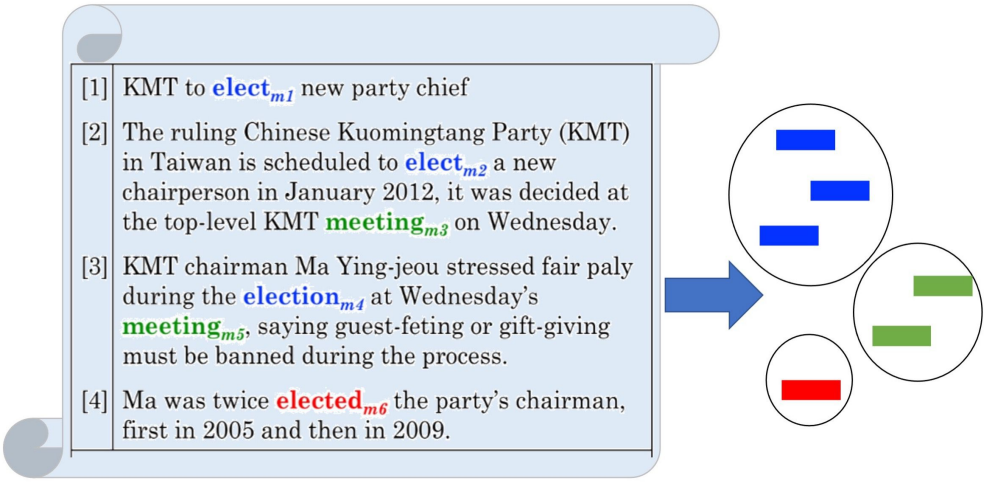


Figure 2.1: Event mentions and their coreferent clusters.

Among these event mentions, m_1, m_2, and m_4 are coreferent, since they all correspond to the same ELECT event of the KMT party electing a new party chief. Similarly, m_3 and m_5 are also coreferent, while m_6 is not coreferent with any other event mention.

Adopting the semantic representation of events (introduced in Section 1.2.1), we model an event mention as a trigger and zero or more corresponding arguments. The trigger of an event mention is the word/phrase that is considered the most representative of the event, such as the word *meeting* for m_3 or the word *elected* for m_6 in Figure 2.1. Arguments are the participants of events, each having its specific argument role. For example, *KMT* is the ENTITY-argument and *new party chief* is the POSITION-argument of m_1.

There are two necessary conditions for two event mentions to be coreferent: trigger relatedness and argument compatibility.

Trigger relatedness Triggers of coreferent event mentions must be semantically related, that is, they should describe the same type of events. For example, m_1 and m_3 cannot be coreferent, since their trigger words — *elect* and *meeting* — are not semantically related. For corpora that incorporate event types, the concept of trigger relatedness is closely related to the type agreement constraint,

Argument role	m_1	m_2	m_4	m_6
PERSON	-	-	-	Ma
ENTITY	KMT	the ruling Chinese Kuomingtang Party (KMT)	-	-
POSITION	new party chief	new chairperson	-	party's chairman
TIME	-	January 2012	-	2005 / 2009

Table 2.1: Event mentions of ELECT type and their arguments. m_1, m_2, and m_4 are coreferent with each other; while m_6 is not coreferent with any other event mentions.

which states that coreferent events must have the same event type. However, trigger relatedness is more strict than the type agreement constraint.

Argument compatibility Argument compatibility is an important linguistic condition for determining the coreferent status between two event mentions. Two arguments are incompatible if they do not correspond to the same real-world entity when they are expressed in the same level of specificity; otherwise, they are compatible. For example, a pair of TIME arguments — *Wednesday* and *2005* — which are expressed in different levels of specificity, are considered as compatible. In Table 2.1, we summarized the arguments of the ELECT event mentions in the previous example (Figure 2.1). If two event mentions have incompatible arguments in any argument role, they cannot be coreferent. For example, m_2 and m_6 are not coreferent since their TIME-arguments — *January 2012* and *2005* — are incompatible. On the other hand, coreferent event mentions can only have compatible arguments. For example, despite the difference in wording, m_1 and m_2 have the same POSITION-arguments (*new party chief* and *new chairperson*). Argument compatibility can serve as a hint suggesting that two event mentions are coreferent. However, we also have to be careful that argument compatibility is not a guarantee for coreference. Take m_1 and m_6 for example, although all the argument roles are compatible, they are still not coreferent.

Coreferent event mentions must obey both trigger relatedness condition and argument compatibility condition. It is relatively simple to model trigger relatedness between event mentions, which can be efficiently captured with even a

moderate size of data. On the other hand, incorporating argument compatibility into event coreference systems is challenging due to the lack of sufficient labeled data. Many existing works relied on the preprocessing of upstream argument extractors and designed heuristic argument features to capture argument compatibility in event coreference resolution systems. However, the error introduced in the preprocessing steps propagates through the event coreference resolution system and hinders the performance considerably.

In light of the aforementioned challenge, we propose a framework for transferring argument (in)compatibility knowledge to the event coreference resolution system, specifically by adopting the interactive inference network [46] as our model structure. The idea is as follows. First, we train a network to determine whether the corresponding arguments of an event mention pair are compatible on automatically labeled training instances collected from a large unlabeled news corpus. Second, to transfer the knowledge of argument (in)compatibility to an event coreference resolver, we employ the network pretrained in the previous step as a starting point and train it to determine whether two event mentions are coreferent on manually labeled event coreference corpora. Third, we iteratively repeat the above two steps, where we use the learned coreference model to relabel the argument compatibility instances, retrain the network to determine argument compatibility, and use the resulting pretrained network to learn an event coreference resolver. In essence, we mutually bootstrap the argument (in)compatibility determination task and the event coreference resolution task.

Our contributions are three-fold. First, we leverage the argument (in)compatibility knowledge acquired from a large unlabeled corpus for event coreference resolution. Second, we employ the interactive inference network as our model structure to iteratively learn argument compatibility and event coreference resolution. Initially proposed for the task of natural language inference, the interactive inference network is suitable for capturing the semantic relations between word pairs. Experimental results on the KBP coreference dataset show that this network architecture is suitable for capturing the argument compatibility between event mentions. Third, our model achieves state-of-the-art results on the KBP 2017 English dataset [40, 41, 44], which confirms the effectiveness of our method.

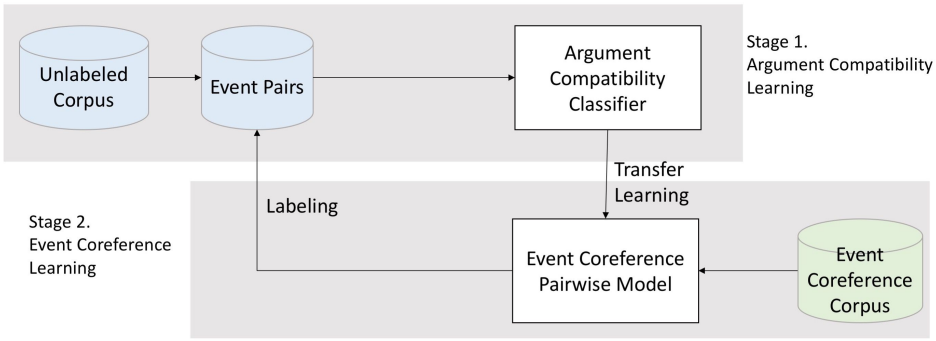


Figure 2.2: System overview.

2.2 Method

Our proposed transfer learning framework consists of two learning stages, the pretraining stage of an argument compatibility classifier and the fine-tuning stage of an event coreference resolver (Figure 2.2). In the pretraining stage, we heuristically extract training samples from large unlabeled corpus and train an **argument compatibility classifier** (Section 2.2.1). In the fine-tuning stage, the argument compatibility classifier is fine-tuned into a **mention-pair event coreference classifier** with the labeled event coreference resolution corpus (Section 2.2.2). In addition, an **iterative relabeling strategy** is used to combine the above two training stages (Section 2.2.3). Last, we introduce the model structure of interactive inference network (IIN) used in both pretraining and fine-tuning stages (Section 2.2.4).

2.2.1 Pretraining Stage: Argument Compatibility Learning

In the pretraining stage, we train a binary argument compatibility classifier which determines whether the arguments between two event mentions are compatible. We heuristically extract training samples from large unlabeled news corpus for argument compatibility learning. In the following, we describe the task of argument compatibility learning and introduce the heuristic method of how we extract training samples from an unlabeled corpus.

Task Description

We define the argument compatibility learning task as follows:

> *Given a pair of event mentions (m_a, m_b) with related triggers, predict whether their arguments are compatible or not.*

Event Representation We do not identify arguments explicitly in this work. In the task definition above, an event mention is represented by a trigger word and the context words within a n-word window around the trigger.

Related trigger extraction Among the two necessary conditions of event coreference resolution, we focus on the argument compatibility condition and assume the input event mentions have related triggers.

We analyze the event coreference resolution corpus and extract trigger pairs that are coreferent more than k times in the training data. We define these trigger pairs to be related triggers in our experiment. Table 2.2 shows some examples of related triggers and the number of times (counts) that the trigger pairs appear as coreferent in the training set of KBP corpus.

trigger pair	count
kill - death	86
attack - strike	61
address - explaination	72
shoot - shooting	35
retire - retire	34
demonstration - protest	30
arrest - custody	13

Table 2.2: Examples of related triggers.

If the triggers of an event mention pair are related, their coreferent status cannot be determined by looking at the triggers alone, and this is the case in which argument compatibility affects the coreferent status most directly. Thus,

	event mention	DATE-compatibility with m_a
m_a	The result of the **election** last October surprised everyone.	-
m_1	He was **elected** as president in 2005.	no
m_2	The presidential **election** took place on October 20th.	yes
m_3	The opposition party won the **election**.	yes

Table 2.3: Examples of NER-based sample filtering. The phrases tagged as DATE are underlined, and the trigger words are boldfaced.

we focus on the event mention pairs with related triggers in the pretraining stage of argument compatibility learning.

Heuristic Training Sample Extraction

From a large unlabeled corpus, we heuristically extract training samples for argument compatibility learning. We collect positive (compatible) samples from event mentions with related triggers in the same document, and apply NER-based filtering to remove bad samples. On the other hand, negative (incompatible) samples consist of event mentions with related triggers from different documents. The details of training sample extraction are summarized in the following:

Compatible samples extraction To collect positive (compatible) samples, we first extract event mention pairs with related triggers within the same document. Further, we apply filtering rules to remove samples that are likely to be problematic. Only event mention pairs that satisfy the following rules are extracted as positive samples:

1. DATE-compatibility (Table 2.3):
   First, we perform named entity recognition (NER) on the context words. If both event mentions have phrases tagged as DATE in the context, these two phrases must contain at least one overlapping word. If there are multiple phrases tagged as DATE in the context, only the phrase closest to the trigger word is considered.

2. PERSON-compatibility: Similar to 1.

3. NUMBER-compatibility: Similar to 1.

4. LOCATION-compatibility: Similar to 1.

5. Apart from function words, the ratio of overlapping words in their contexts must be under 0.3 for both event mentions. We add this constraint in order to remove trivial samples of nearly identical sentences.

Conditions 1–4 are heuristic filtering rules based on NER tags, which aim to remove samples with apparent incompatibilities. Here, we consider four NER types — DATE, PERSON, NUMBER, and LOCATION — because these types of phrases are the most salient types of incompatibility between event mentions.

Condition 5 aims to remove event mention pairs that are "too similar". We add this condition because we do not want our model to base its decisions on the number of overlapping words between the event mentions.

Incompatible sample extraction To collect negative (incompatible) samples, we extract event mentions with related triggers from different documents. Event mention pairs extracted from different documents are almost guaranteed to be not coreferent, and are likely to contain incompatible arguments. We apply the following filtering rules to remove inappropriate negative samples:

1. The creation date of the two documents must be at least one month apart.

2. Apart from the trigger words and the function words, the context of the event mentions must contain at least one overlapping word.

In the unlabeled news corpus, articles describing similar news events are sometimes present. Thus, we use condition 1 to roughly assure that the event mention pairs extracted are not coreferent. Mention pairs extracted from the same document tend to contain overlapping content words, so to prevent our model to make decisions based on the existence of overlapping words, we add condition 2 as a constraint.

We collect event mention pairs satisfying all the above conditions as our initial set of incompatible samples.

Argument compatibility classifier With the initial set of compatible and incompatible samples acquired above, we train a binary classier to distinguish between samples of the two sets.

2.2.2 Fine-tuning Stage: Event Coreference Learning

In the fine-tuning stage, we adapt the argument compatibility classifier on the labeled event coreference data to a mention-pair event coreference model. In the following, we describe the event mention detection method, the mention-pair event coreference model training, and the clustering algorithm.

Event Mention Detection

Before proceeding to the task of event coreference resolution, we have to identify the event mentions in the input documents. We train a separate event mention detection model to identify event mentions along with their subtypes.

We model event mention detection as a multi-class classification problem. Given a candidate word along with its context, we predict the subtype of the event mention triggered by the word. If the given candidate word is not a trigger, we label it as NULL. We select the words that have appeared as a trigger at least once in the training data as candidate trigger words. We do not consider multi-word triggers in this work.

Given an input sentence, we first represent each of its comprising words by the concatenation of the word embedding and the character embedding of the word. These representation vectors are fed into a bidirectional LSTM (biLSTM) layer to obtain the hidden representation of each word.

For each candidate word in the sentence, its hidden representation is fed into the inference layer to predict the class label. Since the class distribution is highly unbalanced, with the NULL label significantly outnumbering all the other labels, we use a weighted softmax at the inference layer to obtain the probability of each class. In this work, we set the weight to 0.1 for the NULL class label and 1 for all the other class labels.

Intuitively, candidate triggers with the same surface form in the same document tend to have the same class label. However, it is difficult to model this

consistency since our model operates at the sentence level. Thus, we account for this consistency across sentences by the following post-processing step: If a candidate word is assigned the NULL label but more than half of the candidates sharing the same surface form is detected as triggers of a specific subtype, then we change the label to this given subtype. Also, we disregard event mentions with types *contact*, *movement* and *transaction* in this post-processing step, since the subtypes under these three types do not have a good consistency across different sentences in the same document.

Mention-Pair Event Coreference Model

With the argument compatibility classifier trained in the previous stage, we use the labeled event coreference corpus to fine-tune the model into an event coreference resolver. We design the event coreference resolver to be a mention-pair model [120], which takes a pair of event mentions as the input and outputs the likelihood of them being coreferent.

Clustering of Event Mentions

With the pairwise event coreference predictions, we further conduct best-first clustering [93] on the pairwise results to build the event coreference clusters of each document. Best-first clustering is an agglomerative clustering algorithm that links each event mention to the antecedent event mention with the highest coreference likelihood given the likelihood is above an empirically determined threshold.

2.2.3 Iterative Relabeling Strategy

Previously, we collected a set of compatible event mentions from the same document with simple heuristic filtering. Despite this filtering step, the initial compatible set is noisy. Here, we introduce an iterative relabeling strategy to improve the quality of the compatible set of event mentions.

First, we calculate the coreference likelihood of the event mentions in the initial compatible set. Mention pairs with a coreference likelihood above threshold θ_M are added to the new compatible set. On the other hand, mention pairs with a

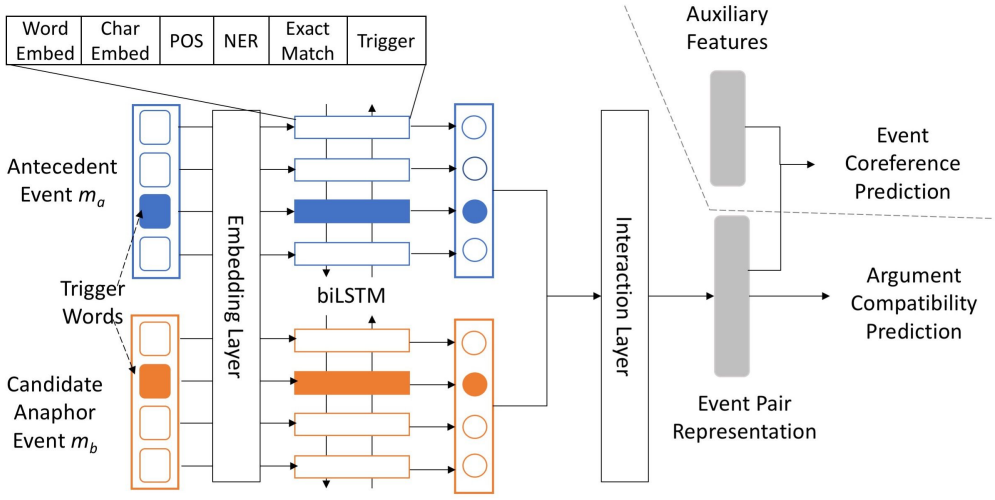


Figure 2.3: Interactive inference network model structure.

coreference likelihood below θ_m are added to the initial incompatible set to form the new incompatible set. With the new compatible and incompatible sets, we can start another iteration of transfer learning to train a coreference resolver with improved quality. In this work, we set θ_M to 0.8 and θ_m to 0.2.

2.2.4 Model Structure

For both the pretraining and fine-tuning stage, we adopt the same interactive inference network as the model structure of our proposed method (Figure 2.3). A qualitative analysis of an interactive inference network shows that it is good at capturing word overlaps, antonyms, and paraphrases between sentence pairs [46]. Thus, we believe this network is suitable for capturing the argument compatibility between two event mentions. The model consists of the following components:

Model inputs The input to the model is a pair of event mentions (m_a, m_b), with m_a being the antecedent mention of m_b:

$$\begin{aligned} m_a &= \{w_a^1, w_a^2, ..., w_a^N\} \\ m_b &= \{w_b^1, w_b^2, ..., w_b^N\} \end{aligned} \tag{2.1}$$

Each event mention is represented by a sequence of N tokens consisting of one trigger word and its context. Here, we take the context to be the words within a n-word window around the trigger.

Embedding layer We represent each input token by the concatenation of the following components:

Word embedding The word representation of the given token. We use pretrained word vectors to initialize the word embedding layer.

Character embedding To identify (in)compatibilities regarding person, organization or location names, the handling of out-of-vocabulary (OOV) words is critical.

Adding character-level embeddings can alleviate the OOV problem [136]. Thus, we apply a convolutional neural network over the comprising characters of each token to acquire the corresponding character embedding.

POS and NER one-hot vectors One-hot vectors of the part-of-speech (POS) tag and NER tag.

Exact match A binary feature indicating whether a given token appears in the context of both event mentions. This feature is proved useful for several NLP tasks operating on pairs of texts [21, 46, 100].

Trigger position We encode the position of the trigger word by adding a binary feature to indicate whether a given token is a trigger word.

Encoding layer We pass the sequence of embedding vectors into a biLSTM layer [50], resulting in a sequence of hidden vectors of size $|h|$:

$$\begin{aligned} h_a^i &= biLSTM(emb(w_a^i), h_a^{i-1}) \\ h_b^i &= biLSTM(emb(w_b^i), h_b^{i-1}) \end{aligned} \tag{2.2}$$

where $emb(w)$ is the embedding vector of token w.

Interaction layer The interaction layer captures the relations between two event mentions based on the hidden vectors h_a and h_b. The interaction tensor I, a 3-D tensor of shape (N, N, $|h|$), is calculated by taking the pairwise multiplication of the corresponding hidden vectors:

$$I_{ij} = h_a^i \circ h_b^j \tag{2.3}$$

Finally, we apply a multi-layer convolutional neural network to extract the event pair representation vector f_{ev}.

Inference layer In the pretraining stage, we feed f_{ev} to a fully-connected inference layer to make a binary prediction of argument compatibility.

As for the fine-tuning stage, we concatenate an auxiliary feature vector f_{aux} to f_{ev} before feeding it into the inference layer. f_{aux} consists of two features, a one-hot vector that encodes the sentence distance between the two event mentions and the difference of the word embedding vectors of the two triggers.

2.3 Evaluation

2.3.1 Experimental Setup

Corpora

We use the English Gigaword corpus [101] as the unlabeled corpus for argument compatibility learning. This corpus consists of the news articles from five news sources, each annotated with its creation date. We extracted 2 million positive and negative samples for the pretraining stage.

As for event coreference resolution corpora, we use the English portion of the KBP 2015 and 2016 datasets [40, 41] for training, and the KBP 2017 dataset [44] for evaluation. The KBP datasets consist of news articles and discussion forum threads. The KBP 2015, 2016, and 2017 corpora contain 648, 169, and 167 documents, respectively. Each document is annotated with event mentions of 9 types and 38 subtypes, along with the coreference clusters of these event mentions. 619,244 pairs of events are used to train the mention-pair coreference model in the fine-tuning stage.

Implementation Details

Preprocessing We use the Stanford CoreNLP toolkit [86] to perform preprocessing on the input data. Each event mention is represented by a trigger word and the context words within a 10-word window. Also, we acquire the list of related triggers by extracting trigger pairs that are coreferent more than 10 times in the training split of the KBP corpus.

Network structure Each word embedding is initialized with the 300-dimensional pretrained GloVe embedding [103]. The character embedding layer is a combination of an 8-dimensional embedding layer and three 1D convolution layers with a kernel size of 5 with 100 filters. The size of the biLSTM layer is 200. The maximum length of a word is 16 characters; shorter words are padded with zero and longer words are cropped. For the interaction layer, we use convolution layers with a kernel size of 3 in combination with max-pooling layers. The size of the inference layer is 128. Sigmoid activation is used for the inference layer, and all other layers use ReLU as the activation function.

Event mention detection model For word embeddings, we use the concatenation of a 300-dimensional pretrained GloVe embedding and the 50-dimensional embedding proposed by Turian et al. [124]. The character embedding layer is a combination of an 8-dimensional embedding layer and three 1D convolution layers with kernel sizes of 3, 4, 5 with 50 filters.

Evaluation Metrics

We follow the standard evaluation setup adopted in the official evaluation of the KBP event nugget detection and coreference task. This evaluation setup is based on four distinct evaluation measures — MUC [128] , B^3 [7], $CEAF_e$ [83] and BLANC [111] — and the unweighted average of their F-scores (AVG-F). We use AVG-F as the main evaluation measure when comparing system performances.

Model	MUC	B^3	$CEAF_e$	BLANC	AVG-F
biLSTM (standard)	29.49	43.15	39.91	24.15	34.18
biLSTM (transfer)	33.84	42.91	38.39	26.59	35.43
Interact (standard)	31.12	42.84	39.01	24.99	34.49
Interact (transfer)	34.28	42.93	39.95	32.12	36.24
Interact (transfer, 2^{nd} iter)	35.66	43.20	40.02	32.43	**36.75**
Interact (transfer, 3^{rd} iter)	36.05	43.07	39.69	28.06	36.72
Jiang et al. (2017)	30.63	43.84	39.86	26.97	35.33

Table 2.4: Event coreference resolution results of our proposed system, compared with the biLSTM baseline model and the previous state-of-the-art system.

2.3.2 Results

We present the experimental results on the KBP 2017 corpus in Table 2.4. In the following, we compare the performance of methods with different network architectures and experimental settings.

Comparison of network architectures We compare the results of the interactive inference network (**Interact**) with the biLSTM baseline model (**biLSTM**).

The biLSTM baseline model does not have the interaction layer. Instead, the last hidden vectors of the biLSTM layer are concatenated and fed into the inference layer directly.

When trained solely on the event coreference corpus (standard), the model with the interactive inference network performs slightly better than the biLSTM baseline model, as shown in rows 1 and 3. However, with an additional pretraining step of argument compatibility learning (transfer), the interact inference network outperforms the biLSTM baseline model by a considerable margin, as shown in rows 2 and 4. We conclude that the interactive inference network can better capture the complex interactions between two event mentions, accounting for the difference in performance.

Effect of transfer learning We compare the results of the model trained only on the event coreference resolution corpus (**standard**) and the model pretrained with the argument compatibility learning task (**transfer**).

Regardless of the network structure, we observe a considerable improvement in performance by pretraining the model as an argument compatibility classifier. The biLSTM baseline model achieves an improvement of 1.25 points in AVG-F by doing transfer learning, as can be seen in rows 1 and 2. As for the interactive inference network, an improvement of 1.75 points in AVG-F is achieved, as can be seen in rows 3 and 4. These results provide suggestive evidence that our proposed transfer learning framework, which utilizes a large unlabeled corpus to perform argument compatibility learning, is effective.

Effect of iterative relabeling We achieve another boost in performance by using the trained event coreference resolver to relabel the training samples for argument compatibility learning. The best result is achieved after two iterations (**Interact(transfer, 2$^{\text{nd}}$ iter)**, row 5) with an improvement of 2.26 points in AVG-F compared to the standard interactive inference network (row 3). However, we are not able to obtain further gains with more iterations of relabeling (**Interact(transfer, 3$^{\text{rd}}$ iter)**, row 6). We speculate that the difference in event coreference model predictions across different iterations is not big enough to have a perceivable impact, but additional experiments are needed to determine the reason.

Comparison with the state of the art Comparing row 5 and 7, we can see that our method outperforms the previous state-of-the-art model [55] by 1.42 points in AVG-F.

2.4 Discussion

In this section, we conduct a qualitative analysis of our proposed event coreference resolution system. First, we analyze the event mentions pairs with related triggers and categorize the argument (in)compatibilities between them into three main categories (Section 2.4.1). Also, we conduct a case study and observe how

our proposed models react to the change of arguments with a swap-argument experiment, in order to verify the effectiveness of our proposed transfer learning framework (Section 2.4.2).

2.4.1 Compatibility Categories

To get a better understanding of argument (in)compatibility, we analyze the event mention pairs with related triggers having either compatible or incompatible arguments. Based on our observations on these event mention pairs, we categorize the argument (in)compatibilities into the following three main categories: explicit, implicit and general-specific (in)compatibilities. Table 2.5 shows the examples of the above (in)compatibility categories and the pairwise prediction results of our best-performing system (the Interact (transfer, 2^{nd} iter) system in Table 2.4). In the following, we discuss the three categories of argument (in)compatibilities and their corresponding examples.

Explicit argument compatibility A pair of event mentions is said to have an explicit argument (in)compatibility if there exist identical/distinct time phrases, numbers, location names, or person names in the contexts of them. Explicit argument (in)compatibility is the most straightforward form of (in)compatibility and is arguably the easiest one to identify.

For these event pairs, the existence of identical/distinct phrases with the same NER type is a direct clue toward deciding their coreferent status. Making use of this nature, we perform filtering on the set of compatible samples acquired from the unlabeled corpus in order to remove samples with explicit incompatibility.

Our model can recognize this type of (in)compatibility with relatively high accuracy. Both examples shown in Table 2.5 are classified correctly.

Implicit argument compatibility We said that a pair of event mentions have implicit argument (in)compatibility if external knowledge is required to resolve their compatible status.

Three examples of implicit argument (in)compatibility are shown in Table 2.5. In the first example, the knowledge that a woman *in her 60s* is generally not

Type	Event Mention Pair	Gold	System
Explicit	m_1: ... the building where 13 people were **killed** will be razed, and a memorial ... m_2: In that case, George Hennard **killed** 23 people at a Luby 's restaurant, ...	non-coref	non-coref
Explicit	m_1: ... Democratic United Party on Wednesday **elected** a new interim leader ... m_2: ... head the party reeling from the narrow defeat in the presidential **election** a month ago	non-coref	non-coref
Implicit	m_1: ... a young woman **protester** was brutally slapped while she was demonstrating ... m_2: ... explain why a women **protester** in her 60s was beaten up by policemen ...	non-coref	coref
Implicit	m_1: She died from a brain hemorrhage on July 10, 2003, ... m_2: ... has denied killing his second wife, whom he says **died** in a car accident.	non-coref	non-coref
Implicit	m_1: Nationwide demonstrations held in France to **protest** gay marriage. m_2: ... to **protest** against the country's plan to legalize same-sex marriage.	coref	coref
General	m_1: ... Connecticut elementary school **shooting** has reignited the debate over gun control. m_2: Gun supporters hold that people, not guns, are to blame for the **shootings**.	non-coref	coref
General	m_1: Industrial accidents have injured and **killed** Foxconn workers, and the company also experienced ... m_2: ... explosion in May 2011 at Foxconn 's Chengdu factory **killed** three workers ...	non-coref	non-coref

Table 2.5: Examples of event pairs with related triggers. Trigger words are boldfaced, and words with (in)compatibility information are colored in blue.

referred to as being *young* is required to determine the incompatibility. Similarly, the knowledge that both *brain hemorrhage* and *car accident* are causes of people's death is required to classify the second example correctly. In the third example, the compatibility of argument *gay marriage* and *same-sex marriage* is supported by the hypernym-hyponym relation of them.

While the performance on samples with implicit (in)compatibility is not as good as that on samples with explicit (in)compatibility, our system is able to capture implicit (in)compatibility to some extent. We believe that this type of (in)compatibility is difficult to be captured with the heuristically designed argument features that are designed based on the outputs of argument extractors and entity coreference resolvers, and that the ability to resolve implicit (in)compatibility contributes largely to our system's performance improvements.

General-specific incompatibility Event mentions describing general events pose special challenges to the task of event coreference resolution. A pair of event mentions is said to have general-specific incompatibility if one event mention describes a general event while the other event mentions describe a specific event.

In Table 2.5, we present two typical examples of this category. In the first example, the second event mention does not refer to any specific shooting event in the real world, in contrast to the first event mention, which describes a specific school shooting event. Similarly, for the second example, where the first event mention depicts a general event and the second event mention depicts a specific one.

General event mentions typically have few or even no arguments and modifiers, making the identification of non-coreference relations very challenging. Since we cannot rely on argument compatibility, a deeper understanding of the semantics of the event mentions is needed. General event mentions account for a considerable fraction of our system's error, since general events are quite pervasive in both news articles and discussion forum threads.

	Event Mention Pair	Type	Proposed	Baseline
I	m_1: What would have happened if Steve Jobs had never **left** Apple ...	-	-	-
	m_2^a: ...in the state that is today if Jobs hadn't **left**.	Explicit	coref	coref
	m_2^b: ...in the state that is today if John hadn't **left**.	Explicit	non-coref	coref
	m_2^c: ...in the state that is today if he hadn't **left**.	Implicit	coref	coref
	m_2^d: ...in the state that is today if she hadn't **left**.	Implicit	non-coref	coref
II	m_1: Police arrest 6 men for **gangraping** housewife in northern India.	-	-	-
	m_2^a: Indian police have arrested six men for allegedly **gangraping** a 29-year-old housewife ...	Explicit	coref	non-coref
	m_2^b: Indian police have arrested six men for allegedly **gangraping** a woman ...	Implicit	coref	non-coref
	m_2^c: Indian police have arrested six men for allegedly **gangraping** a medical student ...	Implicit	non-coref	non-coref
III	m_1: Nationwide demonstrations in France to **protest** gay marriage.	-	-	-
	m_2^a: ...took to the streets across the country to **protest** against the country's plan to legalize same-sex marriage.	Implicit	coref	coref
	m_2^b: ...took to the streets across the country to **protest** against the contentious citizenship amendment bill.	Implicit	non-coref	coref

Table 2.6: Case study on manually-generated event mention pairs. Trigger words are boldfaced, and the target arguments are colored in blue.

2.4.2 Case Study

To better understand the behavior of our system, we perform a case study on manually-generated event pairs. To be more specific, we conduct a 'swap-argument' experiment to verify the effectiveness of our proposed transfer learning framework. For a given pair of event mentions, we first alter only one of the arguments and keep the rest of the content fixed. We then observe the behavior of the system across different variations of the altered argument. Table 2.6 shows three examples of the swap-argument experiment. We also present the argument (in)compatibility type and the system prediction of our best-performing system (the Proposed column) and the model without pretraining (the Baseline column).

Example I In this example, we pick the AGENT-argument as the target and alter the AGENT-argument of the second event mention. The event pair (m_1, m_2^a) is coreferent and the explicit argument compatibility between *Steve Jobs* and *Jobs* also supports the coreferent status. Similarly, the event pair (m_1, m_2^b) is non-coreferent due to the explicit incompatibility between *Steve Jobs* and *John*, and the system's prediction is also non-coreferent. Further, we alter the target argument to *he* (m_2^c), which is an implicit compatible argument with *Steve Jobs*. Finally, we alter the target argument to the pronoun *she* (m_2^d), resulting in an implicit incompatibility in the AGENT argument since the *Steve Jobs* is generally not considered a feminine name.

As can be observed in Table 2.6, our proposed system classifies both the explicit and implicit (in)compatibilities correctly. On the other hand, the baseline model without the pretraining process is not able to distinguish the difference between the different arguments.

Example II In this example, we pick the PATIENT-argument as the target and alter the PATIENT-argument of the second event mention. Our proposed system classifies the event pair (m_1, m_2^a) as coreferent, which is reasonable considering the presence of the explicit compatible arguments *housewife* and *29-year-old housewife*. Further, when we alter the target argument to *woman* (m_2^b), the output of the proposed system is still coreferent. This is consistent with our prediction: the

event mentions are likely to be coreferent judging only from the context of the two event mentions. However, when we alter the target argument to *medical student* (m_2^c), the event pair would become non-coreferent due to the incompatibility between *medical student* and *housewife*. Our proposed system classifies the event pair correctly.

On the other hand, the baseline model without pretraining gives the same prediction regardless of the difference of arguments.

Example III In this example, we pick the REASON-argument as the target and alter the REASON-argument of the second event mention. The event pair (m_1, m_2^a) has a pair of implicit compatible arguments in the REASON-argument role and is likely to be coreferent. In contrast, altering the target argument to *contentious citizenship amendment bill* (m_2^b) would yield a pair of implicit incompatible arguments, and the resulting event pair would become non-coreferent.

Our system classifies both event pairs correctly. Comparing the prediction of our proposed system and the baseline system without the pretraining process, we can verify the effectiveness of our proposed transfer learning framework in learning argument compatibility.

2.5 Related Work

Event Coreference Resolution Ablation experiments conducted by Chen et al. [15] provide empirical support for the usefulness of event arguments for event coreference resolution. Hence, it should not be surprising that, with just a few exceptions [116, 6, 77], argument features have been extensively exploited in event coreference systems to capture the argument compatibility between two event mentions. Basic features such as the number of overlapping arguments and the number of unique arguments, and a binary feature encoding whether arguments are conflicting have been proposed [24, 23, 20]. More sophisticated features based on different kinds of similarity measures have also been considered, such as the surface similarity based on Dice coefficient and the WuPalmer WordNet similarity between argument heads [87, 32, 5, 73, 61]. However, these features are computed

using either the outputs of event argument extractors and entity coreference resolvers [3, 17, 18, 76] or semantic parsers [9, 135, 102] and therefore suffer from serious error propagation issues (see Lu et al. [79]). Several previous works proposed joint models to address this problem [62, 81], while others utilized iterative methods to propagate argument information [73, 27] in order to alleviate this issue. However, all of these methods still rely on argument extractors to identify arguments and their roles.

Transfer Learning Classical supervised learning methods train models only for a single task using a single dataset. Although deep neural models have been applied to many machine learning tasks successfully, training neural models from scratch requires a large amount of task-specific data. Transfer learning is a promising approach to alleviate the scarcity of training data. The key idea of transfer learning is to learn from some additional task(s) with abundant data, then transfer the knowledge to the target task with less data.

Transfer learning has been applied successfully to many NLP tasks. Some transfer learning frameworks utilize large unlabeled data for pretraining. Many of them adopt a language modeling objective to learn distributional embeddings of word, sentences, etc [89, 59, 104, 51, 35]. Due to the lack of a large labeled corpus, transfer learning with labeled pretraining data is not as popular in NLP. Low resource machine translation can benefit from transfer learning by pretraining on other high resource language pair(s) [141]. Min et al. [90] report improved results on two benchmark question answering (QA) datasets by performing transfer learning on a model trained on another large, span-supervised QA dataset. Conneau et al. [30] trained a universal sentence representation with a large natural language inference dataset and performs transfer learning on a wide range of target tasks. In this work, we designed a heuristic method to extract training samples from large unlabeled corpus to learn argument compatibility between event mentions.

After this work was published, Lu et al. [82] proposed an end-to-end event coreference model based on pretrained BERT [35]. They showed that combining the pretrained BERT model with the interactive inference network can produce

superior performance. However, it remains an open question whether it is possible to combine our proposed transfer learning framework with pretrained BERT and give further performance gain, we will leave this as future work.

2.6 Conclusion

We proposed an iterative transfer learning framework for event coreference resolution. Our method exploited a large unlabeled corpus to learn a wide range of (in)compatibilities between arguments, which contributes to the improvement in performance on the event coreference resolution task. We achieved state-of-the-art results on the KBP 2017 English event coreference dataset, outperforming the previous state-of-the-art system. In addition, a qualitative analysis of the system output confirmed the ability of our system to capture (in)compatibilities between two event mentions.

Chapter 3

Narrative Event Relation Knowledge Acquisition from Large Japanese Corpora

In this section, we focus on the task of narrative event relation knowledge acquisition. The narrative event relation knowledge is a type of commonsense knowledge of critical importance which represents the typical co-occurring patterns of events with causal or temporal relations.

In this work, we consider a narrative event pair with shared arguments as a form of narrative event relation knowledge. The shared arguments are the common participants of the narrative related events, and are helpful for NLP tasks such as zero anaphora resolution. We adopt a two-stage framework for Japanese narrative event relation knowledge acquisition. We first extract narrative related event pairs from a large corpus with a statistical method, and then identify the shared arguments between the event pairs. We manually constructed a dataset of narrative related event pairs annotated with shared arguments. We use the annotated corpus to learn and evaluate the share argument identification model.

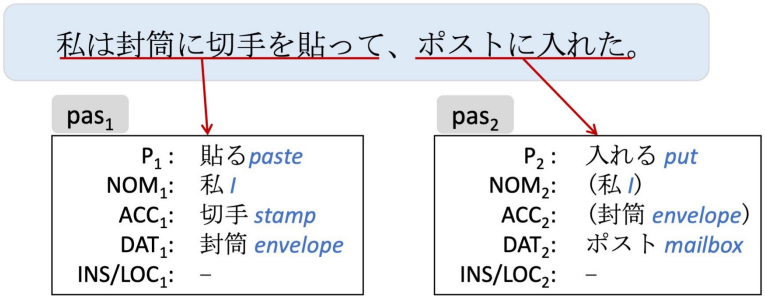


Figure 3.1: Narrative event pair with shared arguments.[1]

3.1 Introduction

As introduced in Section 1.3.1, narrative event relation is an important type of commonsense knowledge that encodes how humans perceive the world around them. Narrative event relation captures the stereotypical ordering of events and covers several types of event relations such as temporal relations, causal relations, etc.

Narrative Event Pairs In this section, we focus on the narrative related event pairs. Figure 3.1 represents an example of narrative event pair, which consists of two events, pas_1 and pas_2. Here, we adopt a statistical point of view of narrative event relation. Event pairs that co-occur more often than expected by chance are defined to be narrative event pairs. Thus, pas_1 and pas_2 are narrative event pairs since we expect pas_2 to follow pas_1 with significant probability under normal situations.

In this work, we adopt the syntactic representation of events (introduced in Section 1.2.1) and define an event as a predicate-argument structure (PAS). An event consists of a predicate and zero or more relevant arguments. Predicates are mostly verbs or verbal phrases, such as *paste* and *put* in the above example. Each

[1]In this work we adopt the Japanese case marker, ga, wo, ni, and de, which roughly corresponds to nominative (NOM), accusative (ACC), dative (DAT), and instrumental/locative (INS/LOC) cases.

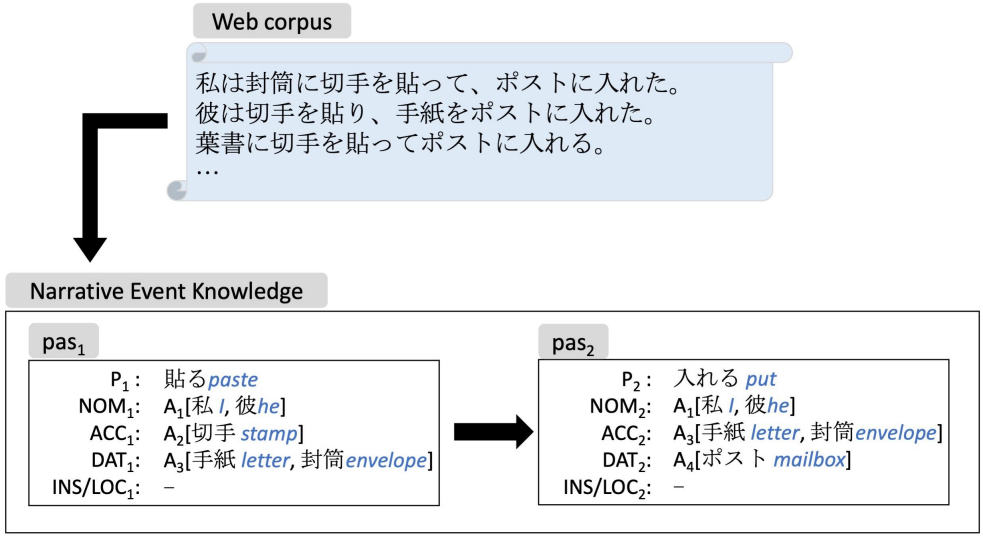


Figure 3.2: Narrative event relation knowledge.

argument is characterized by its syntactic relation with the predicate, and main syntactic markers include nominative (NOM, subject) case, accusative case (ACC, direct object), dative case (DAT, indirect object), and instrumental/locative cases (INS/LOC). For example, *I*, *stamp*, and *envelope* are the nominative, accusative, and dative arguments of pas_1. For pas_2, *mailbox* is the dative argument, *I* and *envelope* are the omitted nominative and accusative arguments of it.

Narrative Event Relation Knowledge In this work, we aim to extract narrative event relation knowledge from large unannotated corpora. A piece of narrative event relation knowledge is in the form of a pair of narrative related events with shared arguments (Figure 3.2). By observing the narrative event pairs in the unannotated corpus, we can further make abstraction of their arguments to represent the common patterns of the arguments. The common patterns of these narrative event pairs are the narrative event relation knowledge we want to extract in this work. Figure 3.2 shows an example of narrative event relation knowledge $pas_1 \Rightarrow pas_2$.

Among the four arguments in $pas_1 \Rightarrow pas_2$, A_1 and A_3 are the shared argu-

ments between the two events, while A_2 and A_4 are arguments that only appears in pas_1 and pas_2, respectively. These shared arguments play an important role in the application of narrative event knowledge since they encode the correspondence relations between case slots within a piece of event relation knowledge. For example, since argument A_3 is shared between the dative case of pas_1 and the accusative case of pas_2, it states the fact that when pas_1 and pas_2 appear successively, the above two case slots should carry the same argument.

On the other hand, for arguments like A_2 and A_4, although these arguments do not encode correspondence relations between case slots, they are also important for specifying the meaning of the predicates. From the perspective of selectional preferences, these arguments put a constraint on the arguments that other case slots can take. For example, if the *stamp* argument (A_2) is in the accusative case of predicate *paste*, them we would expect to see arguments like *letter* or *envelope* in the dative case.

Japanese Narrative Event Relation Knowledge Extraction In this work, we aim to extract the narrative event relation knowledge from Japanese unannotated corpus. Event relation knowledge acquisition in Japanese is a much more challenging task than its counterpart of English, due to several linguistic properties of Japanese. For example:

(c) *John attached a stamp to the letter, and he dropped it into the mailbox.*

(d) *John attached a stamp to the letter, and (ϕ_{he}) dropped (ϕ_{letter}) into the mailbox.*

In the above example, (c) is the Japanese correspondence of (d), directly translated into English. We can observe that Japanese has an abundance of omitted arguments. In addition, Japanese lacks linguistic clues regarding the accordance in gender, number, etc., such as 'he' and 'it' in (c).

These linguistic properties hinder the performance of Japanese coreference resolution systems, and make it unsuitable to apply coreference-based methods of English event relation knowledge acquisition [12] directly to Japanese.

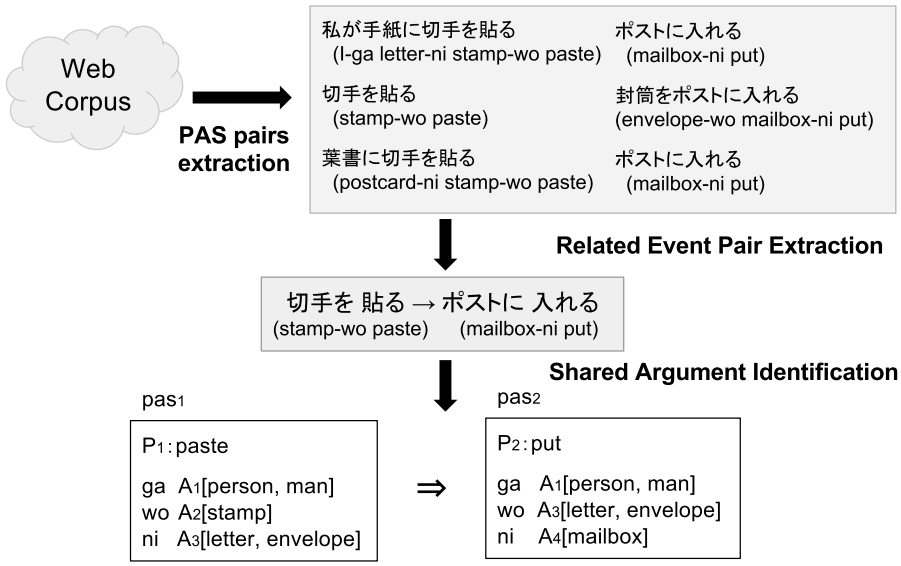


Figure 3.3: Two-stage approach for Japanese event relation knowledge acquisition.

On the other hand, event relation knowledge can benefit the task of the coreference resolution. The shared arguments of narrative event pairs provide direct clues that the case slots sharing an argument should hold co-referring arguments. These clues are particularly critical in cases in which selectional preference is not helpful, such as coreference resolution problems presented in Winograd Schema Challenge [66, 110]. Consider the following example:

(e) *$Google_{NOM}$ acquired $\underline{Motorola_{ACC}}$, because $\underline{they_{NOM}}$ went bankrupt.*

(f) *$A_{1\,NOM}$ go bankrupt → $A_{2\,NOM}$ $A_{1\,ACC}$ acquire*

In the example of (e), both precedents of 'they', 'Google', and 'Motorola', are of the same category. While selectional preference is not helpful in this case, the event relation knowledge in (f) can help us resolve (e) correctly.

Two-stage Framework In this work, we adopt the two-stage framework for Japanese narrative event relation knowledge acquisition proposed in Shibata et al.

[119]. Figure 3.3 shows the system structure of this two-stage approach. In the first stage, narrative event pairs are extracted from large-scale Japanese corpora by association rule mining. In the second stage, we extend the model of Kohama et al. [60] to identify shared arguments of narrative event pairs extracted in the previous stage. We design a richer feature representation for shared argument learning, which considers the interaction between shared arguments and the mechanism of argument omission in depth.

In addition, we manually construct a gold dataset for shared argument learning. With the help of linguistic experts, we established an annotation scheme for shared argument. We train and evaluate our shared argument identification model on the acquired gold dataset. By comparing our proposed methods with several baseline models, we observe a significant improvement in shared argument identification.

3.2 Narrative Event Pair Extraction

We adopt a two-stage framework of narrative event relation knowledge extraction (Figure 3.3). We adopt the first stage of related event pair extraction proposed in their work to obtain the related event pairs, which will be the input to our shared argument identification model.

Starting from the raw web corpus, we extract the predicate-argument pairs with syntactic dependency, and use the Apriori algorithm to pick out the related event pairs efficiently. Also, an additional filtering step is applied to improve the quality of the extracted event pairs, as suggested in Kohama et al..

3.2.1 Event Pair Extraction from Web Corpus

Strongly-related events often appear in the form where they are syntactically dependent with some clause relation. Table 3.1 presents several related event pairs with different clause relation types.

From the web corpus, we first extract all the PAS pairs with syntactic dependency as candidate event pairs. For each PAS, we only keep the arguments of the main surface cases. In this work, we focus on the four main surface cases in

Clause Relation	pas_1	pas_2	Example
Sequence	蜂に刺される	腫れる	蜂に刺されて腫れた。
Contradiction	蜂に刺される	腫れる	蜂に刺されたけど腫れなかった。
Cause	蜂に刺される	腫れる	蜂に刺されたので腫れた。
Condition	蜂に刺される	腫れる	蜂に刺されると腫れる。
Simultaneous	シャワーを浴びる	歯を磨く	シャワーを浴びながら歯を磨く。
Purpose	加熱する	水分を飛ばす	水分を飛ばすために加熱する。

Table 3.1: Narrative event pairs with different type of clause relations.

Japanese: が (ga), を (wo), に (ni), で (de), which roughly correspond to the nominative, accusative, dative, and instrumental/locational cases. Also, attributes such as negation, causative and passive are attached to the predicates.

We keep the order of the event pair ($pas_1 \Rightarrow pas_2$) as they appear in the original text. Only for event pairs with a clause relation of *purpose*, we reverse the order of them in order to indicate the causal relation of the events.

3.2.2 Narrative Event Pair Extraction by Association Rule Mining

After collecting the PAS pairs with syntactic dependency as candidate event pairs, we use the Apriori algorithm to pick out the related event pairs efficiently. Apriori algorithm [11] is an algorithm frequently used for association rule mining. Given a transactional database, along with manually defined thresholds of *support-min*, *confidence-min*, *lift-min* and *lift-max*, the Apriori algorithm finds combinations (rules) that satisfies the co-occurrence conditions.

In this work, we use the Apriori algorithm to extract narrative event pairs from a web corpus of 1.5 billion sentences. By viewing each candidate event pair as a transaction, and the predicates and arguments as items, we can identify narrative event pairs which co-occur with each other with significant statistics. Table 3.2 shows some examples of the narrative event pairs acquired.

Narrative event pair	support($\times 10^7$)	conflict ($\times 10^3$)	Lift
(休養, 仮眠) を取る→抵抗力を高める	1.2	5.6	3170.31
(切手, 印紙) を貼る→ (ポスト, 応募箱) に入れる	1.2	21.4	858.19
(向上, 実現) を目指す→ (研究, 計画) に取り組む	0.8	1.4	221.63
(ウイルス, ノロウイルス) にやられる→ダウンする	0.3	13.8	249.01

Table 3.2: Narrative event pairs acquired by Apriori algorithm.

Type	Narrative event pair	Shared argument
Standard	切手を手紙に貼る→手紙をポストに入れる	n-w
Quasi	牛を飼う→牛乳でチーズを作る	w-d'
Multiple	観光客が町を/に訪れる→町が賑わう	w/n-g

Table 3.3: Types of shared arguments.

3.3 Shared Argument Identification

3.3.1 Gold Dataset

We manually constructed a gold dataset for learning shared argument identification model. In this work, we train and evaluate our proposed model on this gold dataset.

This dataset contains 809 narrative event pairs, with each of the event pairs annotated with its shared argument configuration. Three annotators with linguistic backgrounds participated in the construction of this dataset.

Type of Shared Arguments

The gold dataset contains the following types of shared arguments (Table 3.3):

1. **Standard Shared Argument**:
   The arguments are shared between one case slot of the first event and another case slot of the second event. This type of shared argument represents the fact that arguments of the two cases should correspond to an identical real-world entity.

In this work, we only consider the four main cases of が (nominative, ga), を (accusative, w), に (dative, n), and で (instrumental/locational, d). From now on, we use the shorthand notation of g, w, n, and d to represent these four main cases. The first example in Table 3.3 has a standard shared argument between the first に-case and the second を-case, which both correspond to the entity 'letter'. we use the notation n-w to represent it.

2. **Quasi Shared Argument**:
   Quasi shared arguments consist of a pair of arguments that are closely related to each other in the context of the given event relation knowledge. As can be seen from the example in Table 3.3, the arguments of the first wo-case and the second de-case are 'cow' and 'milk', respectively. These two arguments are considered to be closely related since the milk in the context corresponds to the specific milk which is produced by the cow in the same context.

   We attached an apostrophe (') to denote a quasi shared argument.

3. **Multiple Shared Argument**:
   Multiple shared arguments occur when more than two case slots share the same argument. As can be seen from the example in Table 3.3, the argument 'town' is shared between three cases: wo-case or ni-case of the first event, and the ga-case of the second event.

   We use the symbol '/' to separate different case slots of the same predicate which share arguments.

Preprocessing of Gold dataset

In this work, we only focus on the identification of standard shared arguments. For utilizing the gold dataset with other shared argument types, we perform a pre-processing to the gold annotation before model training. We transform each shared argument configuration into its corresponding standard configuration set.

First, we define the corresponding standard shared argument set for each shared argument in the following manner (Table 3.4):

Type	Shared argument	Standard shared argument set
Standard	n-w	{n-w}
Quasi	w-d'	{w-d, ϕ}
Multiple	w/n-g	{w-g, n-g}

Table 3.4: Transforming different types of shared arguments to their standard shared argument sets.

Shared argument configuration	Standard configuration set
(g-g)	{(g-g)}
(g-g, w-d')	{(g-g, w-d), (g-g)}
(g-g, n-n/w)	{(g-g, n-n), (g-g, n-w)}
(g-g, w-d', n-n/w)	{(g-g, n-n), (g-g, n-w), (g-g, n-n, w-d), (g-g, n-w, w-d)}

Table 3.5: Transforming shared argument configuration to corresponding standard configuration set.

1. For each standard shared argument, we transform it into the standard shared argument set containing only itself.

2. For each quasi shared argument, we transform it into the standard shared argument set containing a null shared argument (ϕ) and its standard counterpart in which all the apostrophe (') mark is removed. See the second example in Table 3.4.

3. For each multiple shared argument, we transform it into the standard shared argument set containing all the shared arguments that could be entailed from it. See the third example in Table 3.4.

For a given shared argument configuration, we first transform each of its containing shared arguments into its corresponding standard shared argument set in the above manner. By taking the product of these standard shared argument sets, we obtain the corresponding standard configuration set of the shared argument configuration. See Table 3.5 for examples.

3.3.2 Case Frame Selection

Selectional preferences provide important clues for the task of share argument identification. Case frames are good sources of selectional preference information, and it handles the issue of predicate ambiguity by clustering the usage of each predicate by their meanings. In turn, the meaning of a case frame is represented by the argument distribution in each case slot of its corresponding case frame.

In this work, we consider wide-coverage case frames constructed automatically from a huge raw corpus as the source of selectional preference information [48]. For each event pair $R(pas_1 \Rightarrow pas_2)$, we select 10 relevant case frames for both pas_1 and pas_2 by utilizing the supporting sentences S of R. Here, we describe the method for selecting relevant case frames for each event pair, which are used in our proposed models.

Given a case frame cf, we denote the bag-of-words (BoW) representation of arguments within each case slot of cf as follows:

$$V^g, V^w, V^n, V^d$$

We denote the BoW representation of arguments appearing in the corresponding case slots of the support sentences S as follows:

$$U^g, U^w, U^n, U^d$$

We define the relevance score of cf with respect to R as follows:

$$rel(cf, R) = \sum_{x=\{g,w,n,d\}} cos(U^x, V^x) \tag{3.1}$$

which is the sum of cosine similarity scores between the BoW representation of case slots in the four main cases.

Finally, we rank all the case frames in descending order with respect to relevance score and take the top 10 of them as relevant case frames. Table 3.6 represents the first five relevant case frames of the predicate 訪れる (*visit*) of the following event pair:

観光客が訪れる→賑わう
(tourist-ga visit → be crowded)

Rank	Case frame	Relevant score
1	(観光客, 人) が (地, 日本) を (実際) に 訪れる	0.966
2	(数人, 人) が (事務所, 京都) を (激励, 視察) に 訪れる	0.807
3	(観光客, 大統領) が (中国, 台湾) を (視察, 見学) に 訪れる	0.760
4	(客, 観光客) が (店, ショップ) を (目当て, 実際) に 訪れる	0.748
5	(人, 観光客) が (博物館, 美術館) を (見学) に 訪れる	0.742

Table 3.6: Relevant case frames of 訪れる (visit).

3.3.3 Joint Prediction of Shared Argument and Case Frame

As mentioned in Section 3.3.2, case frames provide important information of selectional preferences. However, the gold data does not provide the appropriate case frame of each predicate. To tackle this problem, we propose a model of shared argument identification that simultaneously predicts the appropriate case frame for each predicate.

Model

We adopt a maximum entropy (MaxEnt) classifier model.

Given a narrative event pair $R(pas_1 \rightarrow pas_2)$ and its supporting sentences S, the conditional probability of a shared argument configuration $\mathbb{A}$ and case frame pair cf_1, cf_2 is modeled as:

$$P(\mathbb{A}, cf_1, cf_2|R, S; \mathbf{w}) = \frac{exp\{\mathbf{w} \cdot \phi(\mathbb{A}, cf_1, cf_2, R, S)\}}{Z} \tag{3.2}$$

In the above equation, $\phi(\mathbb{A}, cf_1, cf_2, R, S)$ is the feature representation of the shared argument configuration, $\mathbf{w}$ is the model parameter, and Z is the normalization constant. In Table 3.7 we summarized the features used, under the example of shared argument *n-w*.

Feature	description
Configuration	Binary feature indicating the existence of the shared argument n-w
Post-predicate	Binary feature indicating the existence of argument in w case of pas_2
Core	Binary feature indicating if n case of cf_1 and w case of cf_2 are core cases. If a case slot takes argument in more than 10% of the time in the selected case frame, we define it as a core case.
Case slot similarity	The cosine similarity between the vocabulary distribution of n case of cf_1 and w case of cf_2.
Normalized case slot similarity	Case slot similarity of n-w normalized over the similarities of all case slots of cf_1. Same for cf_2.
Conflict	The ratio of support sentences in S that holds different arguments in the first n case and the second w case.
Context	We collect words that appear in Sbut not within the event pair as context words. We calculate the relative probability of each context word to appear in the first n case compared to other main cases, and similar for the second w case. A tf-idf weighted sum of this probability is added as feature.

Table 3.7: Features for shared argument n-w.

Prediction

During the prediction phase, the shared argument configuration $\hat{\mathbb{A}}$ and case frame pair $\hat{cf_1}, \hat{cf_2}$ that gives the highest probability is chosen:

$$(\hat{\mathbb{A}}, \hat{cf_1}, \hat{cf_2}) = \underset{\mathbb{A}, cf_1, cf_2}{\operatorname{argmax}} P(\mathbb{A}, cf_1, cf_2 | R, S; \mathbf{w}) \tag{3.3}$$

For each event pair R, we choose 10 relevant case frames for each predicate of concern as candidate of cf_1 and cf_2, as described in Section 3.3.2.

Model Training

In the training phase, the most probable case frame pair $(\hat{cf_1}, \hat{cf_2})$ and the model parameter $\mathbf{w}$ are updated alternatively. Also, the most probable gold configuration $\hat{g}$ among the standard configuration set is also updated along with the case frame pair.

The training algorithm is summarized below:

1. Initialize model parameter $\mathbf{w}$ randomly.

2. Use the current parameter $\mathbf{w}$ to update the most probable gold configuration and the most probable case frame pair $(\hat{g}, \hat{cf_1}, \hat{cf_2})$:

$$\hat{g}, \hat{cf_1}, \hat{cf_2} = \underset{g, cf_1, cf_2}{\operatorname{argmax}} P(g, cf_1, cf_2 | R, S; \mathbf{w}) \tag{3.4}$$

3. Use $(\hat{g}, \hat{cf_1}, \hat{cf_2})$ to update model parameter $\mathbf{w}$. The following is the objective function, in which the superscripts of g, cf_1, and cf_2 denote the id of the event pairs, and N is the total number of training objects:

$$\begin{aligned} L = & \sum_{n=1}^{N} logP(g^{(n)}, cf_1^{(n)}, cf_2^{(n)} | R, S; \mathbf{w}) \\ & - \alpha \|w\|^2 \end{aligned} \tag{3.5}$$

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmax}} L \tag{3.6}$$

(Hyper-parameter α is set to 1.0.)

4. Back to 2 until convergence. The convergence condition is that the most probable $(\hat{g}, \hat{cf_1}, \hat{cf_2})$ for all event pairs are the same as the previous iteration. If the convergence condition is not satisfied after 15 iterations, we terminate the training process.

3.3.4 Shared Argument Learning with Combined Case Frame

Here, we introduce another model for learning shared arguments that use the combined case frames.

The joint reference model (Section 3.3.3) picks exactly one case frame for each predicate. On the other hand, the combined case frame model combines the relevant case frames by taking the weighted sum of them by the relevance scores with respect to the event pair. This method does not decide the most appropriate case frame of each predicate. Instead, all of the relevant case frames are considered, and case frames with higher relevance scores have a larger influence on the feature representation.

Combined Case Frame

A combined case frame is obtained by combining the relevant case frames according to their relevance scores. The calculation of the relevance scores of each case frame is described in Section 3.3.2.

Given a set of relevant case frames CF, we defined the combined case frame $\widetilde{cf}$ as follows:

$$\widetilde{cf} : \widetilde{V^g}, \widetilde{V^w}, \widetilde{V^n}, \widetilde{V^d} \tag{3.7}$$

$$\widetilde{V^x} = \sum_{cf \in CF} rel(cf, R) \times V^x_{cf}, \forall x \in \{g, w, n, d\} \tag{3.8}$$

in which V^x_{cf} is the vocabulary distribution vector of cf.

Model

Similar to the joint prediction model presented in Section 3.3.3, we adopt a Max-Ent classifier model. Given an event pair $R(pas_1 \Rightarrow pas_2)$ and its supporting sentences S, we model the conditional probability of shared argument configuration $\mathbb{A}$ as:

$$P(\mathbb{A}|R, S; \mathbf{w}) = \frac{exp\{\mathbf{w} \cdot \phi(\mathbb{A}, \widetilde{cf_1}, \widetilde{cf_2}, R, S)\}}{Z} \tag{3.9}$$

In the above equation, ϕ is the feature representation as summarized in Table 3.7, $\mathbf{w}$ is the model parameter, and Z is the normalization constant.

The training algorithm is similar to the one described in Section 3.3.3. In the training phase, the most probable gold configuration $\hat{g}$ and the model parameter $\mathbf{w}$ are updated alternatively until convergence.

3.4 Experiments

3.4.1 Settings

The case frames used in the experiments are built from a web corpus of four billion sentences, with the method proposed by Hayashibe et al. [48].

We use Classias [98] as the implementation of maximum entropy classifier and L-BFGS [97] as the optimization algorithm for learning. We train and evaluate

our proposed models by a 5-fold cross-validation test on the gold shared argument dataset.

3.4.2 Evaluation and Result

We apply three evaluation metrics: precision, recall, and F-score (F_1) for the evaluation of our shared argument identification models.

Model	Precision	Recall	F_1
Baseline[g-g]	0.731	0.717	0.724
Baseline[Kohama+15]	0.729	0.733	0.731
Joint	0.747	**0.786**	**0.766**
Combined	**0.753**	0.748	0.750

Table 3.8: Evaluation result.

We compare our proposed models with two baseline models. The first baseline model, denoted as Baseline[g-g] in Table 3.8, is the majority classifier which gives the output of g-g regardless of the event pair given. The second baseline model, denoted as Baseline[Kohama+15], is the model proposed by Kohama et al. [60].

The experiment results are summarized in Table 3.8. In addition, several event relation knowledge acquired are shown in Table 3.9.

3.4.3 Discussion

Comparison with Baseline Models

As can be observed from Table 3.8, both of our proposed models outperformed the baseline models by a large margin.

Compared to the model proposed by Kohama et al. [60], we use a richer feature representation for shared argument configuration. In their work, a shared argument is represented by the vocabulary distribution similarity between two case slots, such as the similarity between case frames, or the similarity between arguments in the supporting sentences. However, by considering only the distribu-

tional similarities between two case slots, their method overlooked two important intrinsic properties of the shared argument identification task:

1. Interaction of shared arguments:
   Different pieces of shared arguments are not independent, and shared arguments that share a case slot have repulsive effects on each other. For example, if a shared argument configuration already includes *g*-*g*, then it would be unlikely that *g*-*w* also exists in the same configuration. We add the normalized case slot similarity feature which considers not only the case slot similarity of a pair of case slots, but also the relative similarity of them, to account for this property.

2. The mechanism of argument omission in related event pairs:
   High vocabulary distribution similarity indicates the existence of shared arguments, but not vice versa. Consider the following example:
   ジュースが安くなる → ジュースを買う
   (juice-ga become cheaper → juice-wo buy)
   Although there exists a shared argument of *g*-*w*, the vocabulary distributions of the two corresponding case slots are quite different. To address this property, we add the context feature which considers each context word and the relative probability of them to appear in each of the main case slots.

Comparison Between Proposed Models

The major difference between the two proposed models lies in how case frames for feature construction are decided.

As can be observed from Table 3.8, the joint prediction model achieved a better F-score than the combined case frame model. We conclude that deciding the best case frame is a better way for modeling the selectional preference of a predicate, compared to combining case frames with respect to the relevance scores. The result also verified the effectiveness of the joint model of case frames and shared arguments.

Event Pair	Gold Annotation	System Output	Error Type
熟成させる→出荷される (ripen) (ship)	w-g	w-g	-
ジュースが安くなる→買う (juice-ga become cheaper) (buy)	g-w	g-w	-
肌に与える→若返らせる (skin-ni give) (rejuvenate)	w/g-g n-w	g-g n-w	-
切手を貼る→ポストに入れる (stamp-wo paste) (mailbox-ni put)	g-g n-w	g-g	1
迫害される→殺される (suffer persecution) (be killed)	g-g n-n	n-g	2
明るくなる→太陽が顔をだす (become brighter) (sun-ga face-wo appear)	ϕ	g-w	3

Table 3.9: Evaluation results of the proposed and baseline models for shared argument identification.

Error Analysis

In the following are several patterns of error observed in the system output. Examples of each error type are presented in Table 3.9.

1. **Error due to case frame granularity (Error Type 1)**:
   Our proposed model jointly predicts the most appropriate case frame along with the shared argument configuration. By selecting a single case frame for each predicate, we are able to model the selectional preference of the predicates accurately. However, the automatically constructed case frames do not always provide the granularity suitable for our task. If a coarse-grained case frame is selected during the prediction phase, the prediction of shared argument will also be affected.

   For the example shown in Table 3.9, an appropriate case frame of the second predicate 'put' should contain words that support *n-w* shared argument in the wo-case. Table 3.10 represents the most appropriate case frame of the predicate 'put' among all the case frames of this predicate. It can be observed that although the wo-case contains words relevant to the *n-w* shared

Case	Arguments
ga	私, 誰, 人, ママ, 夫, 自分, 母, ..
	(I, who, people, mom, self, mother, ..)
wo	茶, 私, 子供, 花, 模様, .., 手紙, 封筒, ..
	(tea, I, child, flower, pattern, .., letter, .., envelope, ..)
ni	中, 風呂, 部屋, 手, 家, ポスト, ..
	(interior, bathroom, room, hand, house, mailbox)
de	¡数量¿+人, ¡時間¿, 急須, 白, 湯, 鉛筆, ..
	(⟨number⟩+people, ⟨time⟩, teapot, white, hot water, pencil)

Table 3.10: Example of bad case frame

argument, such as 'letter' and 'envelope', there are other irrelevant words dominating this case. These kinds of broad, somewhat noisy case frames hinder the performance of our shared argument identification model.

2. **Error due to event participants with similar characteristics (Error Type 2)**:
   Our method relies largely on the selectional preference information for identifying shared arguments. Thus, the prediction performance of our system is not very good for event pairs containing multiple participants with similar characteristics.

   For the example shown in Table 3.9, our model wrongly identified the shared argument *n-g*. Although both cases are expected to hold human participants, the entity in the first ni-case should correspond to the victim of both actions 'persecute' and 'kill', while the second ga-case should hold the entity of the perpetrator of the two actions. In the scenario of the above event pair, there are two participants of similar characteristics, which are both expected to be human. Since selectional preference cannot effectively distinguish between these similar participants, our model often has difficulty dealing with event pairs with multiple similar participants.

3. **Error due to fixed expression (Error Type 3)**
   In a fixed expression, an argument often takes on a different meaning than

it usually does. Fixed expressions within events sometimes cause problems in shared argument identification. For the example shown in Table 3.9, the system output is as follows:

顔が明るくなる → 太陽が顔を出す

Independently, both PASs shown above are plausible. However, the first PAS, 'face-ga become brighter', means showing a cheerful look; while the second PAS, 'sun-ga face-wo appear', means sun rising. Although both expressions contain the argument 'face', the shared argument of *g-w* does not exist.

3.5 Related Work

As a resource-rich language, coreference resolution of English has achieved a satisfying performance. Thus, several works which utilize coreference information were proposed for English event relation knowledge acquisition.

Chambers et al. [12] introduced the concept of narrative event chains as a representation of structured event relation knowledge. Their method utilizes the coreference chains within the input text to collect events involving the same entity, which they called the *protagonist*. Among the set of events involving the same entity, event sequences that are observed a significant number of times are extracted as typical event sequences.

Pichotta et al. [105] used a richer representation of event than in the work of Chambers et al. and achieved an improvement in predicting performance. Instead of representing an event as a (predicate, dependency) pair, they considered an event as a structure of a predicate and arguments with subject, object, direct object relations with the predicate. With this multi-argument event representation, their model performs better in the cases of ambiguous verbs, and is more capable of capturing complex interactions between multiple entities.

There are several works proposed for Japanese event relation knowledge acquisition utilizing the co-occurrences of events. Abe et al. [1] proposed a pattern-based method which utilized a predefined set of lexico-syntactic co-occurrence patterns to perform bootstrapping for event relation learning. Their work fo-

cused on the acquisition of related event pairs, but not the relations between the arguments of the related events.

Shibata et al. [119] proposed a two-stage approach for Japanese event relation knowledge acquisition (Figure 3.3). In the first stage, related event pairs are extracted from large-scale corpora by association rule mining. In the second stage, shared arguments of the event pairs are identified heuristically based on case slot similarity scores.

Kohama et al. [60] improved the work of Shibata et al. [119] by utilizing crowd-sourced data for shared argument learning. They proposed a joint model that simultaneously predicts the shared argument configuration and disambiguates the meaning of the predicates. However, their work failed to identify the shared arguments accurately for two reasons. First, the crowd-sourced data they used is very noisy and lacks a well-defined standard of labeling. Second, the features used in their model are not sufficient for capturing the characteristics of shared arguments.

3.6 Conclusion

In this work, we proposed a method for shared argument identification in event relation knowledge acquisition. By addressing several problems of the previous works, we improved the shared argument identification model significantly. We proposed a richer feature representation of shared argument configuration which is more suitable for model learning. In order to incorporate different types of shared arguments in the gold dataset, we update the most appropriate gold configuration along with case frames during the training process. We evaluated our model on a manually annotated gold dataset, and our model outperformed the baseline models by a large margin.

Our proposed model jointly predicts the shared argument configuration and the appropriate case frames. By comparing the result of our proposed model with the combined case frame model, we verified the effectiveness of this joint model to predict the appropriate case frames.

Chapter 4

Heterogeneous Graph Based Automatic Summarization

In this section, we focus on the task of automatic summarization and identify the salient contents of a document. We adopt an extractive summarization framework that extracts summary-worthy EDUs to form output summaries. Since EDUs are typically clauses and can be considered as events, our EDU-based extractive summarization framework is in essence identifying the salient events of a document.

For extractive summarization, modeling the relations between text spans in a document is a crucial yet challenging problem. Various kinds of relations exist among text spans of different granularity, such as discourse relations between EDUs and coreference relations between phrase mentions. In this work, we utilize heterogeneous graph to model the various textual relations within a document. The heterogeneous document graph contains three types of nodes, each corresponds to text spans of different granularity. Also, we propose a graph neural network based extractive summarization model that to capture the heterogeneous document graph and the various textual relations in it. Experimental results on benchmark summarization datasets verify the effectiveness of our proposed method.

4.1 Introduction

Automatic summarization is an important NLP task which aims to condense the information of the input document into a shorter summary. The task has two main paradigms: abstractive summarization and extractive summarization. Generating summary sentences from scratch, abstractive summarizers can generate concise and flexible summaries. However, they also suffer from the problem of not being able to reproduce factual details correctly [118]. On the other hand, extractive summarization aims to select salient text spans (mostly sentences) from the input document as summary. Conceptually, extractive summarization is like highlighting the most important parts of the input document. Due to the above nature, extractive summarizers have the advantage of being computationally efficient and factually reliable. In this paper, we will focus on extractive summarization.

Most existing extractive summarization models operate on sentence level, which identify salient sentences from the input document and concatenate them as summary. However, even the extracted sentences may contain redundant details that harm the conciseness of the summary. This kind of sentence-level redundancies exist pervasively even in the salient sentences. Consider the following examples:

(g) ***[Boston native Mark Wahlberg will star in a film about the Boston Marathon bombing and the manhunt,]***$_{EDU_1^a}$ *[Deadline reported Wednesday.]*$_{EDU_2^a}$

(h) ***[Yahya Rashid,]***$_{EDU_1^b}$ *[a UK national from northwest London,]*$_{EDU_2^b}$ ***[was detained at Luton Airport on Tuesday]***$_{EDU_3^b}$ ***[after he arrived on a flight from Istanbul,]***$_{EDU_4^b}$ *[police said.]*$_{EDU_5^b}$

Both sentence (g) and sentence (h) contain important information that represents the central idea of the events being described (highlighted in bold font in the above example). However, they also include peripheral details that could safely be excluded from the summary without harming the overall understanding of the events (colored in gray in the above example).

To avoid sentence-level redundancies, we perform extraction on a smaller granularity of Elementary Discourse Unit (EDU) in this work. EDU is a sub-sentence

unit that originated from discourse analysis [85]. For example, sentence (g) can be further segmented into two EDUs and sentence (h) can be segmented into five EDUs. Among them, only EDU_1^a, EDU_1^b, EDU_3^b and EDU_4^b present central concept of the sentences. By operating on EDU level, our model can eliminate unwanted trivial details like EDU_2^a, EDU_2^b, and EDU_5^b.

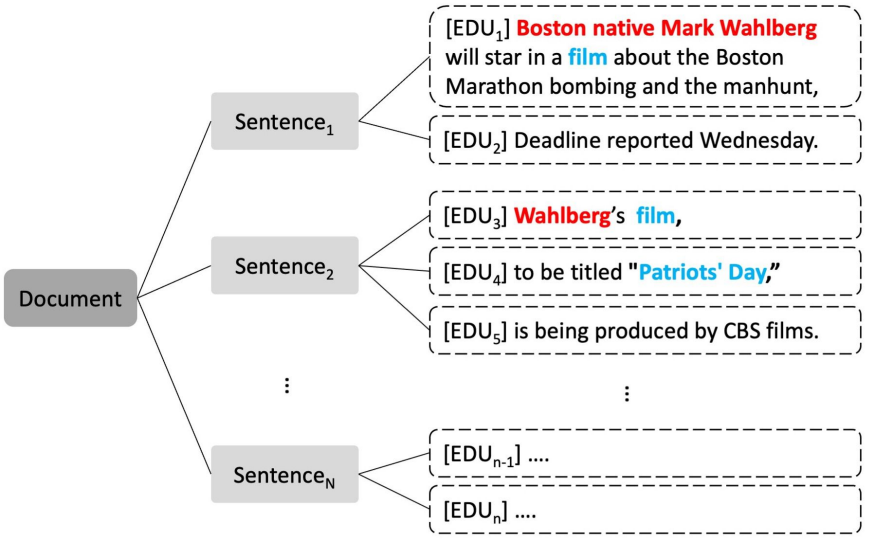


Figure 4.1: Hierarchy document structure.

The goal of extractive summarization is to identify salient text spans that represent the main ideas of the input document. Thus, it is crucial to model the overall document structure and the various relations between the text spans across the document. Natural language documents have a hierarchical nature, with each level corresponding to a different level of granularity: document, sentences, EDUs, words, and phrases (Figure 4.1). Between text spans of different granularity, there exist many different kinds of relations. For example, discourse relations exist between EDUs within a document and coreference relations exist between mention phrases that correspond to the same real-world entity (Figure 4.2). The discourse relations between EDUs provide important clues for the extractive summarization task. The overall discourse structure of a document captures the high-level linguistic structure of the composing events of the document. Also, the nucleus and satellite discourse units directly represent the relative

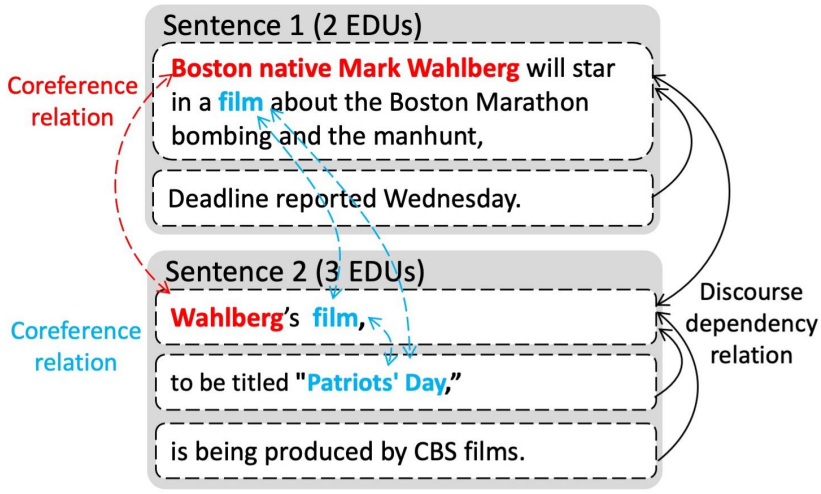


Figure 4.2: Discourse and coreference relations.

saliency of discourse units. Thus, discourse relations are helpful in identifying the salient EDUs among an input document. On the other hand, the coreference relations between the phrase entities are also helpful for the extractive summarization task. Since salient entities are often mentioned multiple times within different contexts in a given document, the coreference relations between the mention phrases can implicitly capture the narrative structure of the document. Figure 4.3 shows a document where the protagonist 'Yahya Rashid' is mentioned multiple times. By observing each mention of the entity 'Yahya Rashid' as well as its context, we can gather information about the narrative centered around the given protagonist.

Due to its complex nature, modeling the various relations among text spans of a document remains an open challenge. Some recent works capture inter-sentential relations by utilizing recurrent neural networks (RNNs) or Transformer [126] based encoders on top of the acquired sentence representations [25, 92, 71]. However, empirical observations show that these sentence-level encoders do not bring much performance gain [71]. Graph structure is an intuitive way to model long-range relations among text spans throughout a document. Early works build connectivity graphs based on content similarity between sentences [42, 88]. Some recent works incorporate discourse or coreference relations into the graph structure and utilize graph neural networks (GNNs) to obtain a high-level representation of

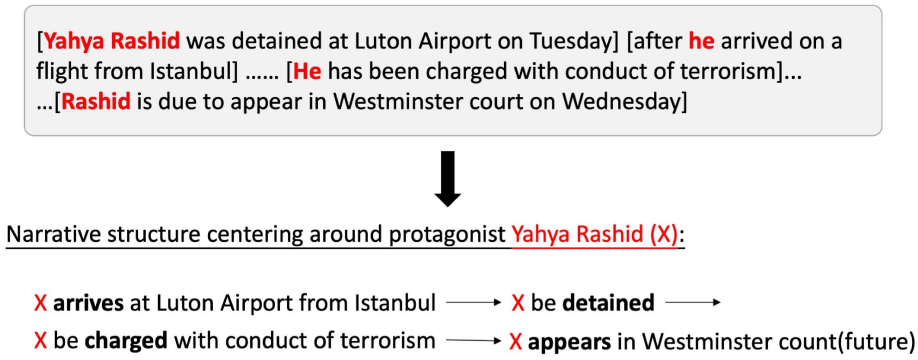


Figure 4.3: Coreference relations around a given entity reflects the narrative structure in which the entity being the protagonist.

text spans [138, 133, 134]. Most of these works operate on homogeneous graphs such as Approximate Discourse Graph (ADG) [29] or Rhetorical Structure Theory (RST) [85] dependency graph. By definition, a homogeneous graph contains only one type of node as well as only one type of edge. On the other hand, a heterogeneous graph consists of more than one type of node or more than one type of edge. As illustrated in Figure 4.1, the various types of relations exist between text spans of different granularity. To model the various text spans of different granularity (nodes) as well as the various types of relations (edges) among them, a heterogeneous graph is a more natural choice than homogeneous graphs.

In this paper, we propose a novel heterogeneous graph based model for extractive summarization. First, we model each input document as a heterogeneous document graph. The heterogeneous graph contains three types of nodes of different granularity: sentence nodes, EDU nodes, and entity nodes. We simultaneously encode both discourse and coreference relations into the graph structure. We encode the discourse relations with the edges between EDU nodes. As for the coreference relations, edges between EDU nodes and entity nodes are introduced. In addition, we model the hierarchical structure of the document by adding edges between sentence nodes and their constituent EDU nodes. Further, we adopt a Graph ATtention network (GAT) [127] based graph encoder to capture the various relations in the heterogeneous document graph. Different from the original

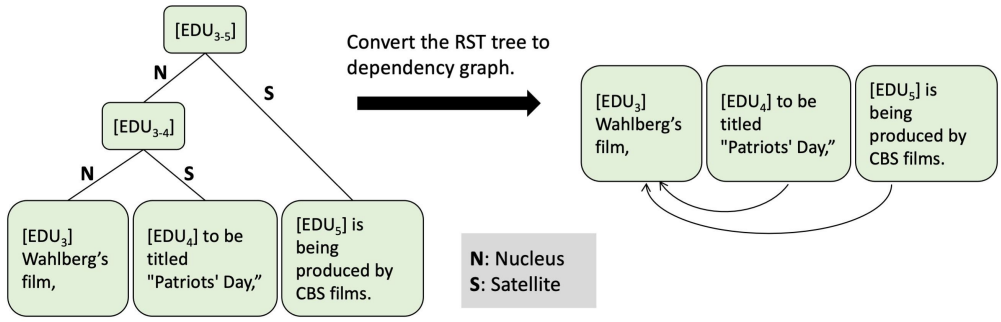


Figure 4.4: An example of RST discourse parse tree. Based on the rhetorical relations among the discourse units, we can derive the RST dependency graph.

GAT network, which is proposed for handling homogeneous graphs, our proposed graph encoder considers the heterogeneous nature of the document graph and applies separate processing for each node type. To the best of our knowledge, we are the first to utilize heterogeneous graphs to incorporate multiple types of relations (discourse relations and coreference relations) simultaneously for extractive summarization.

Our main contribution is threefold: (1) We propose to model the document with a heterogeneous document graph that incorporates multiple types of relations simultaneously for extractive summarization. (2) We propose a GAT-based graph encoder that considers the heterogeneity of the document graph. (3) We conduct experiments on summarization benchmark dataset CNN/DailyMail (CNNDM) [49] and verify the effectiveness of our proposed method.

4.2 Heterogeneous Document Graph

In this section, we introduce the construction of heterogeneous document graphs. Section 4.2.1 gives details of the pre-processing steps to obtain the discourse and coreference information required for the construction of document graphs. Section 4.2.1 gives a brief introduction of the homogeneous document graphs used in previous works as well as their limitations. At last, Section 4.2.2 and 4.2.3 introduce two heterogeneous document graphs which tackle the limitations of the homogeneous document graphs.

4.2.1 Pre-processing

Both discourse information and coreference information are helpful in deciding the content saliency, and have been incorporate as external knowledge into extractive summarization systems.

Given an input document D, we first perform the following pre-processing steps and construct the document graphs based on the discourse information and coreference information acquired.

Discourse Parsing

Given document D with m sentences $\{s_1, ..., s_m\}$, we first segment the sentences into n contiguous, adjacent and non-overlapping EDUs $\{d_1, ..., d_n\}$.

Further, we perform RST discourse parsing to identify the rhetorical relations between the EDUs. In the RST framework, the discourse structure of D is represented in a tree format. For example, Figure 4.4 illustrates the RST parse tree consisting of five EDUs. The RST parse tree is constructed by continuously merging adjacent discourse units to a larger discourse unit. In addition, the merged discourse units are tagged as either nucleus(N) or satellite(S), which indicates their relative nuclearity/saliency. Nucleus units are considered more salient, while satellite units are less important in content. RST defines two types of rhetorical relations between discourse units:

- **Mononuclear relation** that links a satellite unit to a nucleus unit. Such as relation between EDU_1 and EDU_2.
- **Multinuclear relation** that links two nucleus units. Such as the relations between EDU_{1-2} and EDU_{3-5}.

Similar to Xu et al. [134], we convert the RST tree to the dependency form based on the rhetorical relations in the RST tree. The RST dependency graph consists of EDU nodes. In the case of mononuclear relations, the dependency graph contains directed edges from satellite nodes to the corresponding nucleus nodes. In the case of multinuclear relations, we link the nucleus nodes in both directions.

Coreference Resolution

In addition, we perform coreference resolution to identify coreferent relations among mention phrases in the input document D. The goal is to identify mention phrases that refer to the same real-world entity, such as the mentions 'Boston native Mark Wahlberg' and 'Wahlberg' (highlighted in red) in Figure 4.1. The mentions in D are clustered into k entities $\{e_1, ..., e_k\}$, with each entity e_i representing a cluster of mentions among which coreference relations holds.

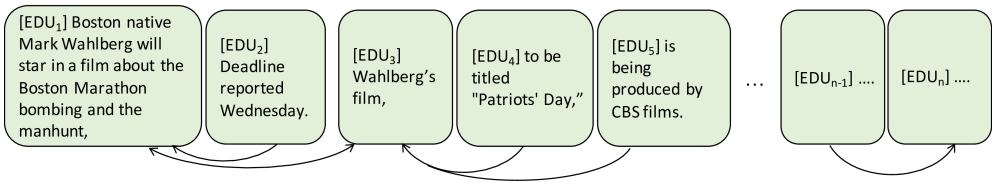
subsectionHomogeneous Document Graphs In many graph-based extractive summarization researches, discourse information and coreference information are used to build homogeneous document graphs with sentence/EDU nodes. By definition, homogeneous graphs are graphs that contain only one type of nodes and one type of edge.

For example, we can embed the discourse relations with the RST dependency graph (Figure 4.4). Also, we can build a homogeneous coreference graph (Figure 4.5(a)) with EDU nodes and EDUs which contain a common entity are linked. However, it is not straightforward to incorporate discourse relations and coreference relations together in a single homogeneous graph.

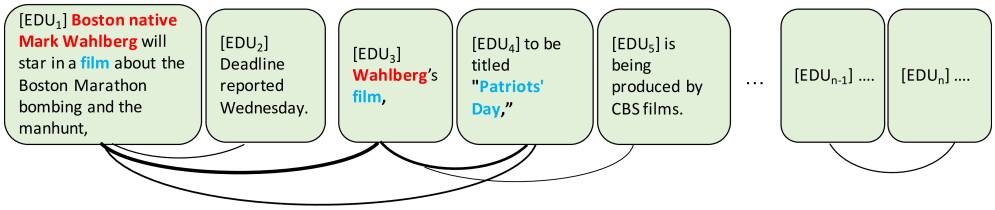
Some previous works like DiscoBERT [134] works use two different graphs to embed discourse and coreference relations separately. However, this method neglects the interaction between different relation types between EDUs. Some other works use weighted graphs (4.5(b)) like ADG to embed multiple types of textual relations, with the edge weights represents the overall 'strength' of relations between text nodes [29]. Although it is possible to combine multiple different relation types within the same graph in this manner, it is difficult to design the proper edge weights.

4.2.2 Heterogeneous Document Graph with Multiple Edge Types

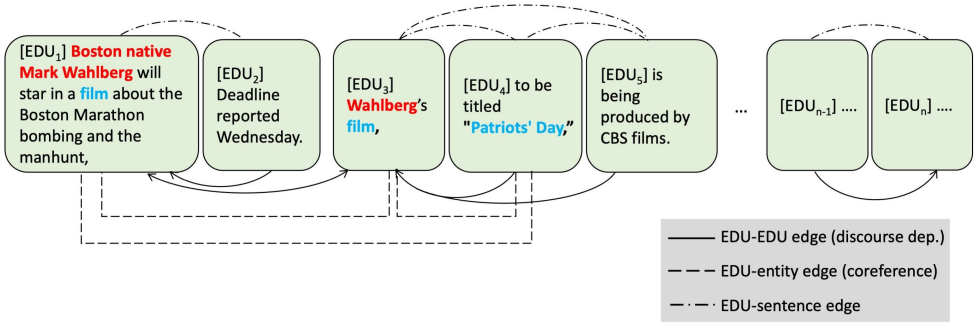
To tackle the limitations of homogeneous document graphs, we consider a heterogeneous document graph with multiple edge types. As shown in Figure 4.5(c), we represent each input document D with a heterogeneous graph $G_1 = \{V, E\}$, with V and E being the set of nodes and edges, respectively.



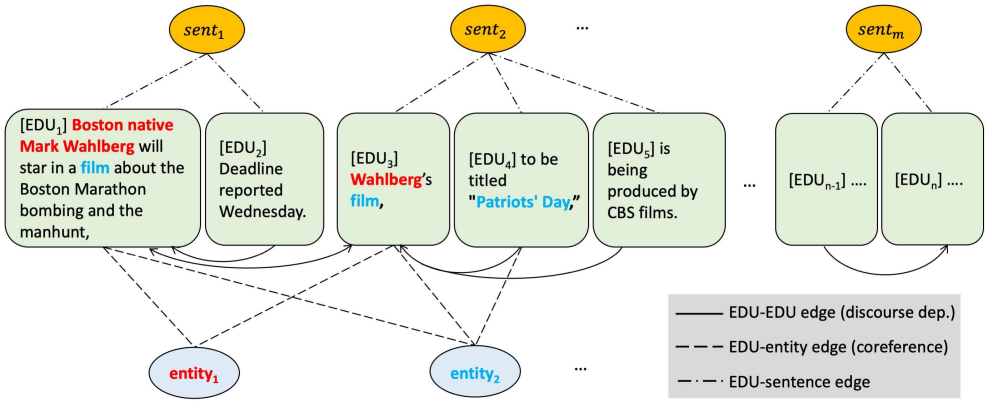
(a) Homogeneous coreference graph.



(b) Weighted homogeneous document graph.



(c) Heterogeneous document graph with multiple edge types (G_1).



(d) Heterogeneous document graph with multiple node types (G_2).

Figure 4.5: Homogeneous and heterogenous document graphs.

In the pre-processing step, we perform discourse parsing and coreference resolution on D (Section 4.2.1). We utilize the pre-processing results to construct the heterogeneous document graph G_1.

$V = \{d_1, ..., d_n\}$ consists of the EDU nodes, with each node d_i representing the i^{th} EDU in the document D.

E consists of three types of edges:

- discourse edge: we add the RST dependency edges between EDU based on the discourse parsing results.
- coreference edge: based on the coreference resolution results, the two EDUs which contain mentions of the same entity are connected.
- same-sentence edge: we connect the EDU nodes that belong to the same sentence.

In this work, we will use the above-mentioned document graph G_1 as a baseline to compare with our proposed heterogeneous document graph with multiple node types (Section 4.2.3).

4.2.3 Heterogeneous Document Graph with Multiple Node Types

Although we can embed the coreference relations with the edges between EDU nodes like in Figure 4.5(a) - (c), the important information of the coreference entity is overlooked. For example, in Figure 4.5(c), a coreference edge only indicates that the two EDUs share one or more common entities, but does not give information regarding what entities actually connect the EDUs.

To tackle the above limitations, we propose a heterogeneous document graph with multiple node types. As shown in Figure 4.5(d), each input document D with a heterogeneous document graph $G_2 = \{V, E\}$, where V and E are the set of nodes and edges, respectively.

V contains three types of nodes:

- $V_s = \{s_1, ..., s_m\}$, with s_i representing the i^{th} sentence in D.
- $V_d = \{d_1, ..., d_n\}$, with d_i representing the i^{th} EDU in D.

- $V_e = \{e_1, ..., e_k\}$, with each e_i representing an unique entity in D.

In the pre-processing step, we perform discourse parsing and coreference resolution on D and build the heterogeneous document graph (Section 4.2.1). We utilize the pre-processing results to construct the heterogeneous document graph G_2 as follows:

- Based on the discourse segmentation result, we connect each sentence node to its constituent EDU nodes in G_2.
- We add these RST dependency edges between EDU units to our document graph G to model the discourse structure of the document.
- We use edges between EDU nodes and entity nodes to embed the coreference relations. If EDU d_i contains a mention of entity e_j, then we add an undirected edge (d_i, e_j) to E. That is, each entity node indirectly connects all EDUs with mentions of the entity. In this way, the subgraph around a specific entity node implicitly models the narrative structure related to the entity.

4.3 Proposed Method

4.3.1 Overview

Problem Formulation

Given an input document D with n EDUs $\{d_1, d_2, ..., d_n\}$, we formulate EDU based extractive summarization as a sequence labeling problem. The model predicts a sequence of binary labels $Y = \{y_1, y_2, ..., y_n\}$, where $y_i = 1$ indicates that the i^{th} EDU, d_i, should be included in the summary.

From the human-written summary, we heuristically obtain the oracle labels $\{y_1^*, y_2^*, ..., y_n^*\}$, which can be used to train our extractive summarization model. Further details will be given in Section 4.4.1. In the inference stage, the model predicts the binary labels for each EDU in the input document. The EDUs with label $y_i = 1$ will be concatenated as the summary.

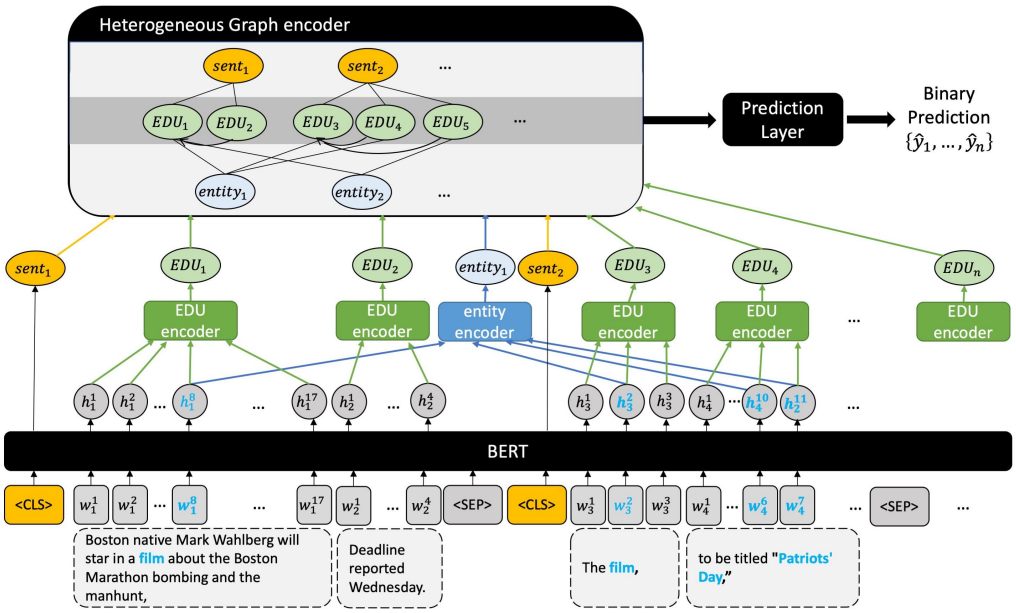


Figure 4.6: System overview.

Model Overview

Figure 4.6 provides an overview of our proposed model. First, the input document D is feed into a Longformer [10] based document encoder. With the self-attention based EDU and entity encoders, we acquire the initial node representation of the heterogeneous document graph (Section 4.3.2). We then apply a heterogeneous graph encoder to obtain high-level node representations (Section 4.3.3). Finally, we make predictions based on the EDU node representation (Section 4.3.4).

4.3.2 Graph Node Initialization

Following the settings in Liu et al. [71], we utilize pretrained Longformer [10] to encode the input document D. We insert the [CLS] and [SEP] special tokens to the beginning and the end of each sentence s_i, respectively. The [CLS] special token was originally used to aggregate features from one or a pair of sentences. Same as Liu et al. [71], we use the [CLS] special token to get representation vectors of the sentence following it. With the output vectors of Longformer, we acquire the initial representations of each node in V as follows:

Sentence Representations

For each sentence node s_i in V_s, we take the Longformer output vector of the [CLS] token before s_i as the sentence node representation h_i^s.

EDU Representations

We use a self-attention based EDU encoder to encode each EDU node in V_d. Given an EDU d_i consisting of tokens $\{w_i^1, w_i^2, ..., w_i^N\}$, we obtain its node representation h_i^d by taking self-attention on the Longformer output vectors $\{v_i^1, v_i^2, ..., v_i^N\}$ of the tokens:

$$\alpha_{ij} = v_2 ReLU(W_1 v_i^j + b_1) \tag{4.1}$$

$$a_{ij} = \frac{exp(\alpha_{ij})}{\sum_{k=1}^{N} exp(\alpha_{ik})} \tag{4.2}$$

$$h_i^d = \sum_j a_{ij} v_i^j \tag{4.3}$$

where W_1, b_1, v_2 are trainable weights.

Entity Representations

The structure of the entity encoder is identical to the EDU encoder. For each entity e_i in V_e, we consider all mentions of it. By taking self-attention among the Longformer output vectors which correspond to tokens of these mentions, we can acquire the entity representation h_i^e.

4.3.3 Heterogeneous Graph Encoder

We initialize the representation of each node in G with the sentence representations ($\{h_i^s\}$), EDU representations ($\{h_i^d\}$), and entity representations ($\{h_i^e\}$) acquired in Section 4.3.2.

Figure 4.7 illustrates the structure of our heterogeneous graph encoder. Each graph encoding layer L_i consists of two Transformer sub-layers and a graph attention sub-layer. The Transformer sub-layers aim to model the interactions among nodes with the same granularity. In the graph attention sub-layer, we apply graph attention network (GAT) [127] to model the interactions among all types of nodes

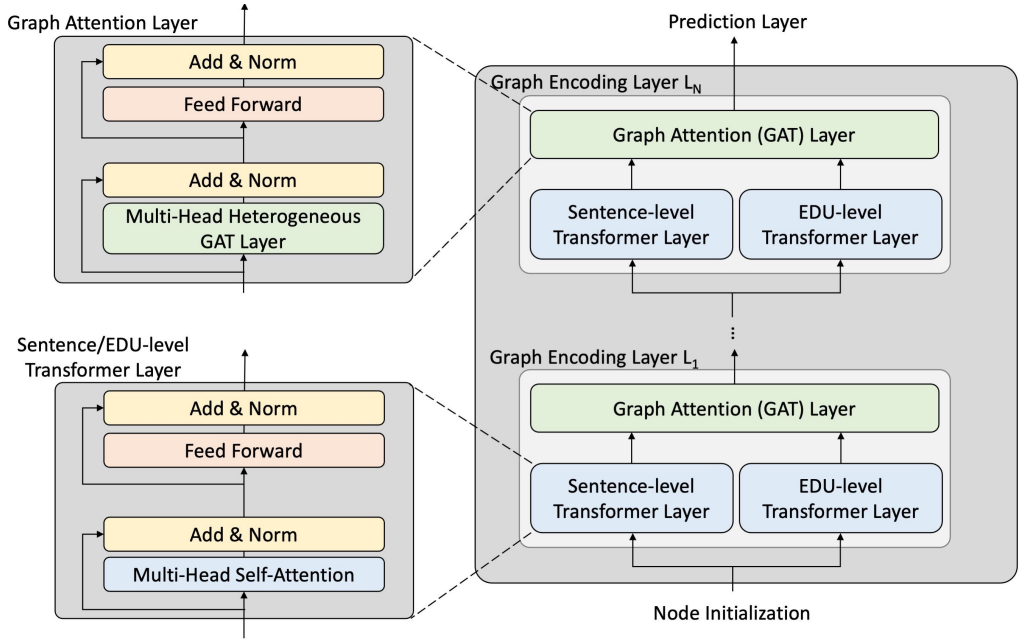


Figure 4.7: Structure of the heterogeneous graph encoder.

in G and update the node representations. The complete heterogeneous graph encoder consists of N stacked graph encoding layer $\{L_1, ..., L_N\}$.

Transformer Sub-layers

We first feed the sentence vectors and EDU vectors to the sentence-level and the EDU-level Transformer [126] sub-layers.

At the center of the Transformer sub-layer is a multi-head self-attention layer follow by a feed-forward layer. The self-attention mechanism can be seen as a fully-connected version of a graph attention network. To model the interactions among nodes of the same granularity, we utilized two types of Transformer layers [126], which operate on the sentence-level and the EDU-level, respectively.

Graph Attention Sub-layer

The graph attention sub-layer consists of a multi-head GAT network followed by a feed-forward layer. Taking the document graph and the node representations as

input, the purpose of the graph attention sub-layer is to learn a higher-level representation of each node by aggregating information from its neighboring nodes. Here, we introduce two types of GAT networks, the **vanilla GAT network** and the **heterogeneous GAT network**. Similar to the one proposed in Veličković et al. [127], the vanilla GAT network handles the document graph G as a homogeneous graph and treats all types of nodes in the same way. On the other hand, our proposed heterogeneous GAT network considers the heterogeneity of G and applies different processing to different node types.

Vanilla GAT network

We apply graph attention networks (GAT) [127] to update the node representations in G. For the i^{th} node, we update the representation h_i of node i with the representations of its neighbors $\{h_j\}$:

$$\alpha_{ij} = LeakyReLU(W_a[W_q h_i; W_k h_j]) \tag{4.4}$$

$$a_{ij} = \frac{exp(\alpha_{ij})}{\sum_k exp(\alpha_{ik})} \tag{4.5}$$

$$h_i \leftarrow W_t(\sigma(\sum_j a_{ij} W_v h_j) + h_i) \tag{4.6}$$

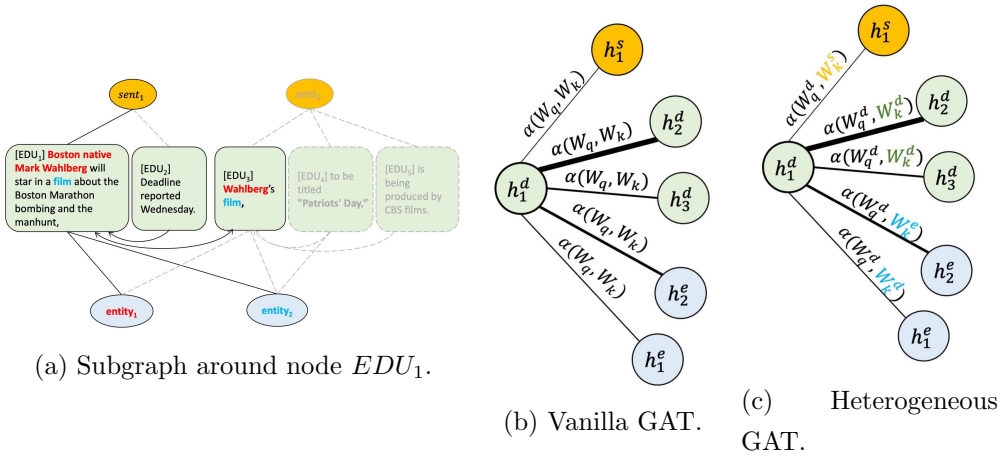
where W_a, W_q, W_k, W_v, W_k are trainable weights.

Figure 4.8 illustrates an example of graph attention mechanism. The subgraph centering around node EDU_1 is highlighted in Figure 4.8(a). EDU_1 is connected to a sentence node $sent_1$, two EDU nodes EDU_2 and EDU_3, and two entity nodes $entity_1$ and $entity_2$. With vanilla GAT network, we calculate the attention weight across the five neighbors of EDU_1 and updated the node representation of EDU_1 (h_1^d) accordingly (4.8(b)).

Although a single GAT network only considers the first-degree neighbors, by stacking several layers of GAT network, we can obtain a higher-level representation for each node in G.

Heterogeneous GAT Network

The vanilla GAT network disregards the heterogeneity of the document graph and treats different types of nodes identically. Figure 4.8(a) illustrates the subgraph



(a) Subgraph around node EDU_1.

(b) Vanilla GAT.

(c) Heterogeneous GAT.

Figure 4.8: Graph attention sub-layer.

of node EDU_1 and its neighbors. All three types of neighboring nodes (sentence, EDU, and entity nodes) use the same equation for calculating attention scores α (Figure 4.8(b)).

Considering the heterogeneous nature of the document graph G_2, we introduce a heterogeneous version of the GAT network (Figure 4.8(c)). To incorporate the multiple node types of heterogeneous document graph G_2, we introduce different query and key matrices for each node type. For example, we use query matrix W_q^d and key matrix W_k^d for EDU nodes, query matrix W_q^e and key matrix W_k^e for entity nodes, and query matrix W_q^s and key matrix W_k^s for sentence nodes.

Different query ($\{W_q^d, W_q^e, W_q^e\}$) and key matrices ($\{W_k^d, W_k^e, W_k^e\}$) are used during the attention calculation process for different types of nodes:

$$\alpha_{ij} = LeakyReLU(W_a[W_q^{t_i} h_i; W_k^{t_j} h_j]) \tag{4.7}$$

$$a_{ij} = \frac{exp(\alpha_{ij})}{\sum_k exp(\alpha_{ik})} \tag{4.8}$$

$$h_i \leftarrow W_t(\sigma(\sum_j a_{ij} W_v h_j) + h_i) \tag{4.9}$$

where t_i indicates the type of node i, and W_a, $\{W_q^d, W_q^e, W_q^e\}$, $\{W_k^d, W_k^e, W_k^e\}$, W_v, W_k are trainable weights.

For example, the attention score α between $sent_1$ and EDU_1 is calculated with the key matrix for sentence nodes W_k^s, while the attention score α between $entity_1$ and EDU_1 is calculated with the key matrix for entity nodes W_k^e.

As for the baseline document graph G_1, we consider a heterogeneous GAT network which considers the multiple edge types. We introduce binary variable d_{ij}, c_{ij} and s_{ij} to indicate the existence of discourse edge, coreference edge and same-sentence edge between node i and node j For example, if there is a discourse edge between node i and node j, then $d_{ij} = 1$, otherwise, $d_{ij} = 0$. The attention weight between node i and node j is calculated as follows:

$$\alpha_{ij} = LeakyReLU(W_a[W_q h_i; d_{ij} W_k^d h_j; c_{ij} W_k^c h_j; s_{ij} W_k^s h_j]) \tag{4.10}$$

4.3.4 Prediction Layer

We feed the final representation of the EDU nodes (h_i^d) to the prediction layer with sigmoid activation to predict binary labels:

$$\hat{y_i} = \sigma(W_p h_i^d + b_p) \tag{4.11}$$

The training loss of the model is the binary cross-entropy loss L against the oracle extraction labels:

$$L = -\sum_i y_i^* log(\hat{y_i}) + (1 - y_i^*) log(1 - \hat{y_i}) \tag{4.12}$$

in which $\{y_i^*\}$ are the oracle labels and $\{\hat{y_i}\}$ are the binary labels predicted by the model.

4.4 Experiment

4.4.1 Experimental Settings

Dataset

We evaluated our proposed model on the benchmark CNN/DailyMail dataset (non-anonymized version) [49]. We used the standard dataset split, which contains 287,227 / 13,368 / 11,490 documents for training, validation, and test split, respectively.

Since the CNN/DailyMail dataset only contains abstractive gold summaries, we have to construct oracle labels heuristically. Similar to Liu et al. [71], we created the oracle labels on EDU-level by greedily selecting EDUs which maximize the ROUGE scores [69]. We used the average of $\text{ROUGE}_1\text{-F}_1$ and $\text{ROUGE}_2\text{-F}_1$ as selection criteria. For each document, we selected up to 5 EDUs.

Pre-processing

We used the Stanford CoreNLP [86] to split sentences. Further, we used the RST discourse parser proposed by Ji et al. [54] for both discourse segmentation and discourse parsing. For coreference resolution, we used the spanBERT-based [56] version of the end-to-end coreference resolver proposed by Lee et al. [65].

Hyper-parameter Settings

We used the 'longformer-base-4096' version of Longformer to encode the input document. The length of each document is truncated to 1024 BPEs. The hidden size of the EDU encoder and the entity encoder is 128. Based on the evaluation losses on the validation set, we set the number of stacked graph encoding layers to $N = 2$. For both Transformer sub-layers and the graph attention sub-layers, the number of attention heads is set to 8, with each head having a hidden size of 96.

Training and Evaluation

During training, we used a batch size of 20. We used Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and followed the learning rate (lr) scheduling in Vaswani et al. [126] with warm-up of 4000 steps (n_{warmup}):

$$lr = 2e^{-3} min({n_{step}}^{-0.5}, n_{step} {n_{warmup}}^{-1.5})$$

All models are trained for 60000 steps. We selected the top-3 checkpoints based on the evaluation losses on the validation set and report the average scores of them on the test set. During the inference phase, the trained model is used to obtain a likelihood score for each EDU. The top-5 EDUs with the highest likelihood scores

Model	R-1	R-2	R-L
LEAD-3	40.34	17.70	36.57
Oracle(Sentence-based)	52.59	31.24	48.87
Oracle(EDU-based)	55.96	34.64	53.26
(Sentence-based extraction)			
BanditSum [38]	41.50	18.70	37.60
NEUSUM [140]	41.59	19.01	37.98
HIBERT [139]	42.37	19.95	38.83
HSG [129]	42.95	19.76	39.23
BertSum (sent) [71]	**43.25**	**20.24**	**39.63**
(EDU-based extraction)			
BertSum (EDU)	42.73	20.03	40.16
DiscoBERT [134]	43.77	**20.85**	40.67
Proposed (vanillaGAT, 768)	43.33	20.55	40.74
Proposed (hetGAT, 768)	43.54	20.63	40.91
Proposed (vanillaGAT, 1024)	43.61	20.67	40.95
Proposed (hetGAT, 1024)	**43.78**	20.81	**41.12**

Table 4.1: Results on the test set of CNN/DailyMail dataset.

are concatenated to generate the final summary. We also perform trigram blocking in the inference phase, which is a simple yet effective way to reduce redundancy in extractive summarization.

We adopt ROUGE [69] as the evaluation metrics. We report the F_1 scores of the $ROUGE_1$, $ROUGE_2$, and $ROUGE_L$ metrics of our proposed models.

4.4.2 Results and Analysis

Results on CNN/DailyMail Dataset

Table 4.1 shows the results on CNN/DailyMail dataset. The first part contains the **LEAD-3** baseline and **Oracle** upper bounds. The second part of the table includes the sentence-based extractive models, and the third part includes the

EDU-based extractive models. In the third part of the table, we present the evaluation scores of our proposed models with **vanillaGAT** and the **hetGAT** version. heterogeneous document graph G_2 (introduced in Section 4.2.3) is used here. Also, we included results of maximum input sizes 1024 and 768. For the input size 768, we use BERT [35] as the encoder instead of Longformer, which is the same setting as DiscoBERT [134].

As Table 4.1 shows, all our proposed models outperform the **LEAD-3** and all sentence based extractive baseline models. Compared to the **BertSum(sent)** baseline, our proposed model (**hetGAT, 1024**) achieved a higher ROUGE on all three metrics (R-1/R-2/R-L). We conclude that EDU-based extraction is a promising direction in extractive summarization.

Our proposed model (**hetGAT**) also outperform the **BertSum(EDU)** baseline by a significant margin in all three metrics (R-1/R-2/R-L). This result shows the effectiveness of our graph encoder module to capture the complex relations among the text spans of the input documents.

Compared to the state-of-the-art EDU-extraction model **DiscoBERT**, our proposed model (**hetGAT, 1024**) achieved comparable performance on R-1/R-2 metrics and outperformed it on the R-L metrics by 0.45 of F_1 score. However, the performance of the proposed model with 768 maximum input size (**hetGAT, 768**) has worse performance compare to DiscoBERT on R-1/R-2 metrics. **DiscoBERT** incorporates a strict RST-based rule during oracle construction and post-processing to ensure discourse consistency. Since the purpose of this paper is to propose a heterogeneous graph based method for modeling text span relations, we will leave the question of discourse consistency to future work. [1]

Compared to DiscoBERT, our proposed model can better adapt to long input documents. First, the homogeneous structure of their work is not efficient in embedding coreference relations. Take the example of an entity with k mentions, the homogeneous coreference graph (like Figure 4(a)) needs to use $\frac{k(k-1)}{2}$ edges, while our proposed heterogeneous graph only needs k edges to represent the coreference

[1] The SpanBERT based end-to-end coreference resolver (F1=0.77, OntoNote corpus) has a better performance than the Stanford CoreNLP coreference resolver (F1=0.69, OntoNote corpus) used in DiscoBERT.

Model	R-1 (ΔR-1)	R-2 (ΔR-2)	R-L (ΔR-L)
Proposed (hetGAT)	43.78	20.81	41.12
- discourse	43.45(**-0.33**)	20.55(**-0.26**)	40.83(**-0.29**)
- coref	43.57(-0.21)	20.68(-0.13)	40.93(-0.19)
- sent	43.74(-0.04)	20.79(-0.02)	41.06(-0.06)
Proposed (vanillaGAT)	43.61	20.67	40.95
- discourse	43.42(**-0.19**)	20.54(**-0.13**)	40.77(**-0.18**)
- coref	43.51(-0.10)	20.61(-0.06)	40.85(-0.10)
- sent	43.57(-0.04)	20.66(-0.01)	40.90(-0.05)

Table 4.2: Ablation studies on CNN/DailyMail dataset.

relation. Since both GCN (used in DiscoBERT) and GAT have time complexity $O(|E|)$, the larger number of edges will make it difficult to adapt to longer documents. Also, unlike their GCN-based method, the GAT-based method we adopted is exempted from eigen-compositions and costly matrix operations.

Finally, for both types of input length (768 and 1024), we can observe that the **hetGAT** model outperforms the **vanillaGAT** model. This shows the effectiveness of our proposed heterogeneous GAT networks to captures and aggregates the various text relations in the heterogeneous document graphs.

Ablation Study

We conduct ablation studies on the components of our proposed model (Table 4.2) with the heterogeneous document graph G_2.

The first part shows the ablation study of the proposed (**hetGAT**) model. First, we remove the RST dependency edges between EDU nodes (**-discourse**). Next, we remove the coreferential edges between EDU nodes and entity nodes (**-coref**). We can see that both discourse and coreference information contributes significantly to the model performance, with discourse information being slightly more important than the coreference information. We also try to remove the edges between sentence nodes and their constituent EDU nodes (**-sent**). However, linking the sentence and EDU nodes does not seem to have a significant impact

Model	R-1	R-2	R-L
Baseline (G1, vanillaGAT)*	43.43	20.58	40.77
Baseline (G1, hetGAT)*	43.53	20.64	40.85
Proposed (G2, vanillaGAT)*	43.61	20.67	40.95
Proposed (G2, hetGAT)*	**43.78**	**20.81**	**41.12**

Table 4.3: Results on different graph structure.

on model performance.

The second part of Table 4.2 shows the ablation study of the proposed (**vanillaGAT**) model. The results of the ablation study show a similar tendency compared to the **hetGAT** model. We can observe from the results that both discourse and coreference information contributes to the model performance separately, but these two different types of information do not aggregate well compared to the case in the **hetGAT** model. This result illustrates the effectiveness of our proposed heterogeneous GAT network in handling various types of text relations simultaneously, compared to the vanilla GAT networks proposed for homogeneous graphs.

Results of Different Graph Structure

We perform experiments on different graph structures. We compare the system performance on the baseline document graph G_1 (Section 4.2.2) and our proposed heterogeneous document graph G_2 (Section 4.2.3).

As shown in Table 4.3, for both **vanillaGAT** and **hetGAT**, using document graph G_2 gives a better performance then using G_1. Although both G_1 and G_2 embeds the discourse, coreference and same-sentence information within the document graph, G_2 includes richer information of the coreferent entities and sentences. The experimental results show the effectiveness of introducing extra entity and sentence nodes in the document graph.

Results of Different Hyper-parameter Settings

We perform experiments on different hyper-parameter settings to observe how the model performance changes under the changes of hyper-parameters.

Figure 4.9: Result of different number of stacked graph encoding layers (N) on validation set.

Same as ablation study, we use heterogeneous graph G_2 in the following experiments.

Number of stacked graph encoding layer We modify the number of stacked graph encoding layers (N) and observe how it affects the performance. We report the performance on the validation set from $N = 1$ to $N = 3$ in terms of validation loss, ROUGE-1, ROUGE-2 and ROUGE-L in Figure 4.9.

As can be observed in Figure 4.9, we can see that there is a significant performance gap between $N = 1$ and $N = 2$. On the other hand, there is no significant difference in model performance from $N = 2$ to $N = 3$.

The phenomenon can be explained by the theory of meta-path of the document graph. Meta path is a widely used concept proposed to model the various types of relations between nodes in a heterogeneous graph. A meta path captures a specific type of relations within the given graph. Figure 4.10 illustrates the various meta paths in our proposed heterogeneous document graph:

(a) **EDU-EDU meta path** represents the discourse relation between discourse

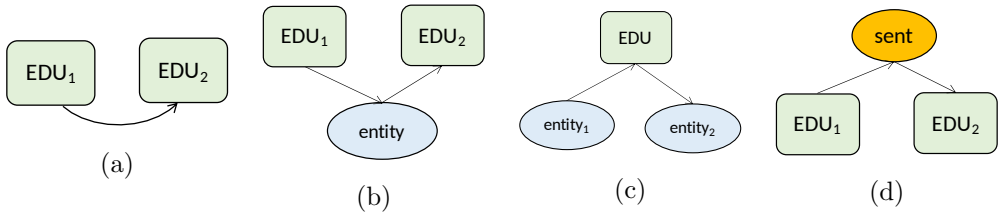


Figure 4.10: Meta paths in the heterogeneous document graph.

units.

(b) **EDU-entity-EDU meta path** connects two EDU nodes sharing a coreferent entity. This 2-hop meta path describes the coreference relation of the entity in the center.

(c) **entity-EDU-entity meta path** describes the collocation relation between two entities that appear in the same EDU unit.

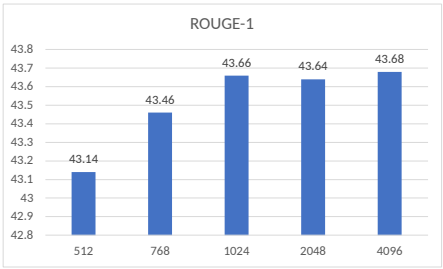
(d) **EDU-sent-EDU meta path** represents the hierarchical structure of a sentence and its consisting EDUs.

The $N = 1$ model cannot capture the 2-hop meta-paths like EDU-entity-EDU, entity-EDU-entity and EDU-sent-EDU. We speculate that this could account for the performance loss of the $N = 1$ model.

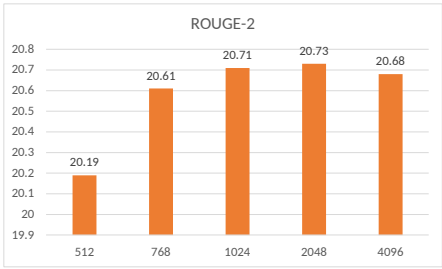
Maximum Input Length We modify the maximum input length (in BPEs) and observe how it affects the performance. We report the ROUGE-1, ROUGE-2 and ROUGE-L scores on the validation set with the maximum input size set to 512, 768, 1024, 2048, 4096 BPEs in Figure 4.11.

Generally, we can observe a performance gain by increasing the maximum input length. The performance gain is significant for the maximum input size under 1024. However, the gain is less significant if we increase the input size further.

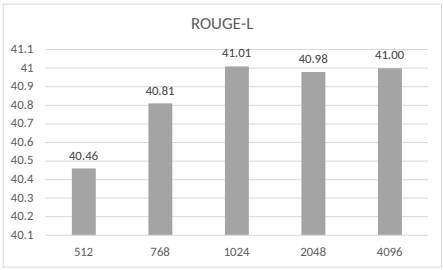
The average document length of the CNN/DailyMail dataset is around 864 BPEs, with a standard deviation of 443 BPEs. Also, consider the fact that summary-worthy, salient sentences tend to appear at the beginning of the docu-



(a)



(b)



(c)

Figure 4.11: Relations among text spans of different granularity.

ment. The above two facts altogether support that setting the maximum input length to 1024 BPEs should give satisfactory results.

Qualitative Analysis

We also conduct a qualitative analysis of the proposed model. The effectiveness of discourse relations is more straightforward and widely studied in previous research. Thus, we focus on the analysis of the role of coreference information in our proposed summarization model.

In the heterogeneous document graph, EDUs containing the same entity phrase are indirectly connected through the node of the given entity. By analyzing the output of the full proposed model and the model without coreference information (**-coref**), we found that the models rank the importance of coreferent EDUs differently. Table 4.4 indicates a common pattern of the improved cases by incorporating coreference information. The table shows examples of coreferent EDUs and the ranking of their likelihood scores to be included in the summary. Com-

EDUs with coreferent entity	$\text{Rank}_{\text{coref}}$	$\text{Rank}_{\text{w/o coref}}$	Oracle Label
Mexican state oil company Pemex said 45 workers were injured ...	1	2	1
... two of them are in serious condition	2	1	0
Kim was accused of stabbing U.S. ambassador Mark Lippert ...	1	3	1
Before Lippert was supposed to give a speech, the attacker slashed him in the face a jaw	3	1	0
Kim stabbed Lippert with a 10-inch knife	2	2	0

Table 4.4: Qualitative studies on CNN/DailyMail dataset.

paring the EDU ranking of the full model ($\mathbf{Rank_{coref}}$) and the model without coreference information ($\mathbf{Rank_{w/o\ coref}}$), we argue that the model with coreference information is better in discriminating the important EDUs among all EDUs sharing the same entity.

4.5 Related Work

Graph-based Summarization

Graph-based summarization models have been broadly explored. Early works build connectivity document graphs based on inter-sentential similarity [42, 88]. With the promising results of graph neural networks (GNNs) [58, 127], some recent works utilize GNN to incorporate external knowledge into the model. For instance, Yasunaga et al. [138] utilizes a sentence-level ADG graph to model discourse and coreference relations. Some works convert the RST tree of the input document into dependency form in either sentence or EDU level [133, 134]. Most of these models operate on homogeneous graphs with only one type of node. Fewer summarization models operate on heterogeneous graphs with different types of nodes. Wei [131] introduces a heterogeneous graph of sentence, word, and topic nodes, and Wang et al. [129] also utilizes a heterogeneous graph of sentence and word nodes. However, neither of the above works incorporates external knowledge into the graph.

Cui et al. [31] perform sentence-based extractive summarization based on the heterogeneous sentence node and nodes representing latent topics.

EDU-based Extractive Summarization

Li et al. [67] illustrate the potential of using EDU as the extraction unit for summarization. Xu et al. [134] also introduce an end-to-end EDU-based extractive summarization model. By using a heuristic based on RST dependency structure, they enhanced the grammaticality and discourse consistency of the extracted summary.

4.6 Conclusion

In this work, we proposed a novel heterogeneous graph based model for extractive summarization. By introducing nodes of different granularity, the heterogeneous document graph has the capacity to embed various types of relations between text spans. In addition, we proposed a heterogeneous GAT network that considers the heterogeneous nature of the document graph. Experiments on the CNN/DailyMail benchmark dataset illustrated the effectiveness of our proposed method.

Chapter 5

Conclusion

5.1 Overview

In this thesis, we study the events in natural language texts. Among the numerous event-related tasks, we focus on three tasks that analyze the different aspects of events: event coreference resolution, narrative event relation knowledge acquisition, and extractive summarization considering heterogeneous event graphs.

In Chapter 2, we focus on the task of **event coreference resolution**. Event coreference resolution is an important subtask of information extraction (IE), the task aims to identify and cluster the event mentions that refer to the same real-world event. In this work, we adopt the semantic view of events by representing events by a trigger and zero or more arguments of different semantic roles. For event mentions to be coreferent, their triggers must be semantically related and their arguments must be compatible. While argument compatibility is an important aspect to determine the coreference status of event mentions, it is difficult to learn argument compatibility due to the limited size of existing annotated corpora. To tackle the above problem, we propose a transfer learning framework of event coreference resolution and learn argument compatibility from the abundant unannotated corpora available. We conducted experiments on the benchmark KBP corpora and verified the effectiveness of our proposed method. A further case study also illustrates the ability of our proposed model to capture the (in)compatibilities of event arguments.

In Chapter 3, we focus on the task of **narrative event relation knowledge acquisition**. The narrative event relation captures the stereotypical ordering of events that are temporally and causally related, which is an important type of commonsense knowledge that reflects the way the human mind perceives the world. In this work, we mine pairs of narratively related events from a large Japanese unannotated corpus. In order to efficiently mine the underlying narratively related events from large unannotated corpus, we take a syntactic view of events by representing each event as a predicate-argument structure (PAS). We first model the event pairs and their consisting predicate/arguments as transactions and items, and utilize association rule mining algorithm to identify narratively related events of statistical significance. Further, considering the abundance of omitted arguments in Japanese texts, we manually constructed an annotated corpus to learn the patterns of the shared arguments of the event pairs. With the 2-stage method above, we collected a large amount of narrative event relation knowledge (400,000 pairs of narrative events) from a large web corpus.

In Chapter 4, we focus on the task of **extractive summarization**. Extractive summarization is one of the main paradigms of the automatic summarization task, which aims to identify the salient parts of a document as summary. In this work, we consider the elementary discourse units (EDUs) as events and formulate extractive summarization as event saliency identification problem. In order to identify the salient and summary-worthy EDUs of the input document, it is critical to model the interactions between text units. First, discourse relations between EDUs capture the high-level linguistic structure of the input document. Also, the nucleus-satellite relations between rhetorically related discourse units also give direct clues of event saliency. Second, coreference relations also play a role to consolidate the information scattered across the input document. For instance, the coreference relations between phrases mentions of a specific entity give information about the narrative structure around the specific protagonist. We utilize heterogeneous graph to model the input document and the textual relations in it. Also, we propose a graph attention (GAT) based graph encoder to capture the structure of the heterogeneous document graph. We conducted experiments on the benchmark CNN/Daily Mail corpus and verified the effectiveness of our

proposed method.

We believe the various event-related tasks presented in this thesis provide a thorough study of the events in natural language texts from multiple aspects. We hope that this thesis will invoke new research ideas for future event-related research.

5.2 Future Work

5.2.1 Event Extraction and Event Coreference Resolution

Event extraction and event coreference resolution are fundamental tasks of information extraction. However, most works on event extraction focus on closed-domain settings and rely on predefined event schemas. The domain-specific predefined event schemas are not easily extended from one domain to other domains. Thus, the development of domain-adaptive event extraction frameworks is an important direction of future studies. Transfer learning and domain adaption methods could be possible solutions.

Event coreference resolution is also a fundamental event-related task. Despite its importance, the performance of the existing is still not good enough for practical use. One of the main problems is the data scarcity problem. In this thesis, we proposed a transfer learning framework of event coreference resolution and utilized the large unlabeled corpus to learn argument compatibility patterns. In future studies, transfer learning or semi-supervised learning methods are likely to remain a major way to tackle the data scarcity problem. Also, most of the existing event coreference resolution frameworks are based on the pairwise/local information and overlook the important overall structure of the document. Incorporating the idea of event saliency (such as main events) and document discourse structure would be a promising future direction of event coreference resolution research.

5.2.2 Global Structure of Events and Event Relations

Modeling the various event relations as well as the global structure of events in natural language texts is a critical task. In Chapter 3, we studied the narrative

event relation between pairs of events; in Chapter 4, we use heterogeneous graph to capture the various relations between events within a document. How to model the various textual relations is still an open question. Previous studies models the narrative relations with typical sequences of events, the discourse theory RST captures the discourse document structure with a tree. Graph is a promising way to model the event structure in documents, which is capable of capturing the complex linguistic structure of natural language texts. With the recent advance of the graph neural network, we expect more future works to explore the possibility of using graph structure to model complex event relations.

Bibliography

[1] S. Abe, K. Inui, and Y. Matsumoto. Acquiring event relation knowledge by learning cooccurrence patterns and fertilizing cooccurrence samples. *Proceedings of the 3rd International Joint Conference on Natural Language Processing*, pages 479–504, 2008.

[2] D. Ahn. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, Sydney, Australia, July 2006. Association for Computational Linguistics.

[3] D. Ahn. The stages of event extraction. In *Proceedings of the COLING/ACL Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, 2006.

[4] J. Allan. *Topic Detection and Tracking: Event-Based Information Organization*, volume 1. 01 2002.

[5] J. Araki, Z. Liu, E. Hovy, and T. Mitamura. Detecting subevent structure for event coreference resolution. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 4553 – 4558, 2014.

[6] J. Araki and T. Mitamura. Joint event trigger identification and event coreference resolution with structured perceptron. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2074–2080, 2015.

[7] A. Bagga and B. Baldwin. Algorithms for scoring coreference chains. In *Proceedings of the LREC Workshop on Linguistics Coreference*, pages 563–566, 1998.

[8] C. Bejan and S. Harabagiu. A linguistic resource for discovering event structures and resolving event coreference. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, May 2008. European Language Resources Association (ELRA).

[9] C. A. Bejan and S. Harabagiu. Unsupervised event coreference resolution. *Computational Linguistics*, 40(2):311–347, 2014.

[10] I. Beltagy, M. E. Peters, and A. Cohan. Longformer: The long-document transformer. *arXiv:2004.05150*, 2020.

[11] C. Borgelt and R. Kruse. Induction of association rules: A priori implementation. *15th Conference on Computational Statistics*, 09 2002.

[12] N. Chambers and D. Jurafsky. Unsupervised learning of narrative event chains. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics*, pages 789–797, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[13] N. Chambers and D. Jurafsky. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 976–986, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[14] C. Chen and V. Ng. Joint modeling for Chinese event extraction with rich linguistic features. In *Proceedings of COLING 2012*, pages 529–544, Mumbai, India, Dec. 2012. The COLING 2012 Organizing Committee.

[15] C. Chen and V. Ng. Chinese event coreference resolution: Understanding the state of the art. In *Proceedings of the 6th International Joint Conference on Natural Language Processing*, pages 822–828, 2013.

[16] C. Chen and V. Ng. SinoCoreferencer: An end-to-end Chinese event coreference resolver. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4532–4538, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA).

[17] C. Chen and V. Ng. SinoCoreferencer: An end-to-end Chinese event coreference resolver. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 4532–4538, 2014.

[18] C. Chen and V. Ng. Chinese event coreference resolution: An unsupervised probabilistic model rivaling supervised resolvers. In *Proceedings of Human Language Technologies: The 2015 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1097–1107, 2015.

[19] C. Chen and V. Ng. Joint inference over a lightly supervised information extraction pipeline: Towards event coreference resolution for resource-scarce languages. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1), Mar. 2016.

[20] C. Chen and V. Ng. Joint inference over a lightly supervised information extraction pipeline: Towards event coreference resolution for resource-scarce languages. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 2913–2920, 2016.

[21] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, 2017.

[22] Z. Chen and H. Ji. Graph-based event coreference resolution. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing (TextGraphs-4)*, pages 54–57, Suntec, Singapore, Aug. 2009. Association for Computational Linguistics.

[23] Z. Chen and H. Ji. Graph-based event coreference resolution. In *Proceedings of the ACL-IJCNLP Workshop on Graph-based Methods for Natural Language Processing (TextGraphs-4)*, pages 54–57, 2009.

[24] Z. Chen, H. Ji, and R. Haralick. A pairwise event coreference model, feature impact and evaluation for event coreference resolution. In *Proceedings of the RANLP Workshop on Events in Emerging Text Types*, number 3, pages 17–22, 2009.

[25] J. Cheng and M. Lapata. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, Berlin, Germany, Aug. 2016. Association for Computational Linguistics.

[26] P. K. Choubey and R. Huang. Event coreference resolution by iteratively unfolding inter-dependencies among events. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2124–2133, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics.

[27] P. K. Choubey and R. Huang. Event coreference resolution by iteratively unfolding inter-dependencies among events. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2124–2133, 2017.

[28] P. K. Choubey and R. Huang. Improving event coreference resolution by modeling correlations between event coreference chains and document topic structures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 485–495, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[29] J. Christensen, Mausam, S. Soderland, and O. Etzioni. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1163–1173, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[30] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes. Supervised learning of universal sentence representations from natural language inference data, 2018.

[31] P. Cui, L. Hu, and Y. Liu. Enhancing extractive text summarization with topic-aware graph neural networks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5360–5371, Barcelona, Spain (Online), Dec. 2020. International Committee on Computational Linguistics.

[32] A. Cybulska and P. Vossen. Semantic relations between events and their time, locations and participants for event coreference resolution. In *Proceedings of the 9th International Conference on Recent Advances in Natural Language Processing*, pages 156–163, 2013.

[33] A. Cybulska and P. Vossen. Using a sledgehammer to crack a nut? lexical diversity and event coreference resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4545–4552, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA).

[34] A. Cybulska and P. Vossen. Translating granularity of event slots into features for event coreference resolution. In *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 1–10, Denver, Colorado, June 2015. Association for Computational Linguistics.

[35] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[36] X. Ding, K. Liao, T. Liu, Z. Li, and J. Duan. Event representation learning enhanced with external commonsense knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the*

9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4894–4903, Hong Kong, China, Nov. 2019. Association for Computational Linguistics.

[37] G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal, May 2004. European Language Resources Association (ELRA).

[38] Y. Dong, Y. Shen, E. Crawford, H. van Hoof, and J. C. K. Cheung. BanditSum: Extractive summarization as a contextual bandit. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3739–3748, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics.

[39] J. Dunietz and D. Gillick. A new entity salience task with millions of training examples. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 205–209, Gothenburg, Sweden, Apr. 2014. Association for Computational Linguistics.

[40] J. Ellis, J. Getman, D. Fore, N. Kuster, Z. Song, A. Bies, and S. Strassel. Overview of linguistic resources for the TAC KBP 2015 evaluations: Methodologies and results. In *Proceedings of the TAC KBP 2015 Workshop*, 2015.

[41] J. Ellis, J. Getman, N. Kuster, Z. Song, A. Bies, and S. Strassel. Overview of linguistic resources for the TAC KBP 2016 evaluations: Methodologies and results. In *Proceedings of the TAC KBP 2016 Workshop*, 2016.

[42] G. Erkan and D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res. (JAIR)*, 22:457–479, 2004.

[43] L. Gao, P. K. Choubey, and R. Huang. Modeling document-level causal structures for event causal relation identification. In *Proceedings of the 2019*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 1808–1817, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[44] J. Getman, J. Ellis, Z. Song, J. Tracey, and S. Strassel. Overview of linguistic resources for the TAC KBP 2017 evaluations: Methodologies and results. In *Proceedings of the TAC KBP 2017 Workshop*, 2017.

[45] G. Glavaš and J. Šnajder. Constructing coherent event hierarchies from news stories. In *Proceedings of TextGraphs-9: the workshop on Graph-based Methods for Natural Language Processing*, pages 34–38, Doha, Qatar, Oct. 2014. Association for Computational Linguistics.

[46] Y. Gong, H. Luo, and J. Zhang. Natural language inference over interaction space. In *Proceedings of the 6th International Conference on Learning Representations*, 2018.

[47] M. Granroth-Wilding and S. Clark. What happens next? event prediction using a compositional neural network model. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2727–2733. AAAI Press, 2016.

[48] Y. Hayashibe, D. Kawahara, and S. Kurohashi. Japanese case frame construction considering varieties of case patterns. Technical report, 2015.

[49] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. Teaching machines to read and comprehend. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1693–1701. Curran Associates, Inc., 2015.

[50] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[51] J. Howard and S. Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for*

Computational Linguistics (Volume 1: Long Papers), pages 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[52] B. Jans, S. Bethard, I. Vulić, and M. F. Moens. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 336–344, Avignon, France, Apr. 2012. Association for Computational Linguistics.

[53] H. Ji and R. Grishman. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*, pages 254–262, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[54] Y. Ji and J. Eisenstein. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13–24, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[55] S. Jiang, Y. Li, T. Qin, Q. Meng, and B. Dong. SRCB entity discovery and linking (EDL) and event nugget systems for TAC 2017. In *Proceedings of the TAC KBP 2017 Workshop*, 2017.

[56] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020.

[57] Jun-Tae Kim and D. I. Moldovan. Acquisition of linguistic patterns for knowledge-based information extraction. *IEEE Transactions on Knowledge and Data Engineering*, 7(5):713–724, 1995.

[58] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[59] R. Kiros, Y. Zhu, R. Salakhutdinov, R. S. Zemel, A. Torralba, R. Urtasun, and S. Fidler. Skip-thought vectors, 2015.

[60] S. Kohama, T. Shibata, and S. Kurohashi. Argument alignment learning in event knowledge. In *Proceedings of the 21th Annual Meeting of the Association for Natural Language Processing (In Japanese)*, pages 1065–1067, Kyoto, Japan, March 2015. Association for Natural Language Processing.

[61] S. Krause, F. Xu, H. Uszkoreit, and D. Weissenborn. Event Linking with Sentential Features from Convolutional Neural Networks. In *Proceedings of the 20th Conference on Computational Natural Language Learning*, pages 239–249, 2016.

[62] H. Lee, M. Recasens, A. Chang, M. Surdeanu, and D. Jurafsky. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 489–500, 2012.

[63] I.-T. Lee and D. Goldwasser. Feel: Featured event embedding learning. pages 4840–4847, 2018.

[64] I.-T. Lee and D. Goldwasser. Multi-relational script learning for discourse relations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4214–4226, Florence, Italy, July 2019. Association for Computational Linguistics.

[65] K. Lee, L. He, M. Lewis, and L. Zettlemoyer. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics.

[66] H. Levesque, E. Davis, and L. Morgenstern. The winograd schema challenge. In *13th International Conference on the Principles of Knowledge Representation and Reasoning*, pages 552–561. Association for the Advancement of Artificial Intelligence, 2012.

[67] J. J. Li, K. Thadani, and A. Stent. The role of discourse units in near-extractive summarization. In *Proceedings of the 17th Annual Meeting of the*

Special Interest Group on Discourse and Dialogue, pages 137–147, Los Angeles, Sept. 2016. Association for Computational Linguistics.

[68] P. Li, Q. Zhu, H. Diao, and G. Zhou. Joint modeling of trigger identification and event type determination in Chinese event extraction. In *Proceedings of COLING 2012*, pages 1635–1652, Mumbai, India, Dec. 2012. The COLING 2012 Organizing Committee.

[69] C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[70] X. Liu, Z. Luo, and H. Huang. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics.

[71] Y. Liu and M. Lapata. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China, Nov. 2019. Association for Computational Linguistics.

[72] Z. Liu, J. Araki, E. Hovy, and T. Mitamura. Supervised within-document event coreference using information propagation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4539–4544, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA).

[73] Z. Liu, J. Araki, E. Hovy, and T. Mitamura. Supervised within-document event coreference using information propagation. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 4539–4544, 2014.

[74] Z. Liu, C. Xiong, T. Mitamura, and E. Hovy. Automatic event salience identification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1226–1236, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics.

[75] J. Lu and V. Ng. Event coreference resolution with multi-pass sieves. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3996–4003, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA).

[76] J. Lu and V. Ng. Event coreference resolution with multi-pass sieves. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, pages 3996–4003, 2016.

[77] J. Lu and V. Ng. Joint learning for event coreference resolution. In *Proceedings of 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 90–101, 2017.

[78] J. Lu and V. Ng. Learning antecedent structures for event coreference resolution. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 113–118, 2017.

[79] J. Lu and V. Ng. Event coreference resolution: A survey of two decades of research. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 5479–5486, 2018.

[80] J. Lu, D. Venugopal, V. Gogate, and V. Ng. Joint inference for event coreference resolution. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3264–3275, Osaka, Japan, Dec. 2016. The COLING 2016 Organizing Committee.

[81] J. Lu, D. Venugopal, V. Gogate, and V. Ng. Joint inference for event coreference resolution. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3264–3275, 2016.

[82] Y. Lu, H. Lin, J. Tang, X. Han, and L. Sun. End-to-end neural event coreference resolution. *CoRR*, abs/2009.08153, 2020.

[83] X. Luo. On coreference resolution performance metrics. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 25–32, 2005.

[84] S. Lv, F. Zhu, and S. Hu. Integrating external event knowledge for script learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 306–315, Barcelona, Spain (Online), Dec. 2020. International Committee on Computational Linguistics.

[85] W. C. Mann and S. A. Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281, 1988.

[86] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[87] K. McConky, R. Nagi, M. Sudit, and W. Hughes. Improving event coreference by context extraction and dynamic feature weighting. In *Proceedings of the 2012 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support*, pages 38–43, 2012.

[88] R. Mihalcea and P. Tarau. TextRank: Bringing order into texts. In *Proceedings of EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing*, July 2004.

[89] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA, 2013. Curran Associates Inc.

[90] S. Min, M. Seo, and H. Hajishirzi. Question answering through transfer learning from large fine-grained supervision data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2:*

Short Papers), pages 510–517, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[91] P. Mirza, R. Sprugnoli, S. Tonelli, and M. Speranza. Annotating causality in the TempEval-3 corpus. In *Proceedings of the EACL 2014 Workshop on Computational Approaches to Causality in Language (CAtoCL)*, pages 10–19, Gothenburg, Sweden, Apr. 2014. Association for Computational Linguistics.

[92] R. Nallapati, F. Zhai, and B. Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 3075–3081. AAAI Press, 2017.

[93] V. Ng and C. Cardie. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 104–111, 2002.

[94] T. Nguyen, A. Meyers, and R. Grishman. New york university 2016 system for kbp event nugget: A deep learning approach. *Theory and Applications of Categories*, 2016.

[95] T. H. Nguyen, K. Cho, and R. Grishman. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309, San Diego, California, June 2016. Association for Computational Linguistics.

[96] T. H. Nguyen and R. Grishman. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 365–371, Beijing, China, July 2015. Association for Computational Linguistics.

[97] J. Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

[98] N. Okazaki. Classias: a collection of machine-learning algorithms for classification, 2009.

[99] T. Otake, S. Yokoi, N. Inoue, R. Takahashi, T. Kuribayashi, and K. Inui. Modeling event salience in narratives via barthes' cardinal functions. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1784–1794, Barcelona, Spain (Online), Dec. 2020. International Committee on Computational Linguistics.

[100] B. Pan, Y. Yang, Z. Zhao, Y. Zhuang, D. Cai, and X. He. Discourse marker augmented network with reinforcement learning for natural language inference. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 989–999, 2018.

[101] R. Parker, D. Graff, J. Kong, K. Chen, and K. Maeda. English Gigaword fourth edition. In *Linguistic Data Consortium*, 2009.

[102] H. Peng, Y. Song, and D. Roth. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 392–402, 2016.

[103] J. Pennington, R. Socher, and C. Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

[104] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[105] K. Pichotta and R. Mooney. Statistical script learning with multi-argument events. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 220–229, Gothenburg, Sweden, 2014. Association for Computational Linguistics.

[106] K. Pichotta and R. Mooney. Statistical script learning with recurrent neural networks. In *Proceedings of the Workshop on Uphill Battles in Language Processing: Scaling Early Achievements to Robust Methods*, pages 11–16, Austin, TX, Nov. 2016. Association for Computational Linguistics.

[107] K. Pichotta and R. J. Mooney. Using sentence-level LSTM language models for script inference. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 279–289, Berlin, Germany, Aug. 2016. Association for Computational Linguistics.

[108] S. S. Pradhan, L. Ramshaw, R. Weischedel, J. MacBride, and L. Micciulla. Unrestricted coreference: Identifying entities and events in ontonotes. In *International Conference on Semantic Computing (ICSC 2007)*, pages 446–453, 2007.

[109] J. Pustejovsky, P. Hanks, R. Saurí, A. See, R. Gaizauskas, A. Setzer, D. Radev, B. Sundheim, D. Day, L. Ferro, and M. Lazo. The timebank corpus. *Proceedings of Corpus Linguistics*, 01 2003.

[110] A. Rahman and V. Ng. Resolving complex cases of definite pronouns: The winograd schema challenge. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 777–789, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

[111] M. Recasens and E. Hovy. BLANC: Implementing the Rand Index for coreference evaluation. *Natural Language Engineering*, 17(4):485–510, 2011.

[112] E. Riloff. Automatically constructing a dictionary for information extraction tasks. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, AAAI'93, page 811–816. AAAI Press, 1993.

[113] E. Riloff and J. Shoen. Automatically acquiring conceptual patterns without an annotated corpus. In *Third Workshop on Very Large Corpora*, 1995.

[114] A. Romadhony, D. Widyantoro, and A. Purwarianti. Utilizing structured knowledge bases in open ie based event template extraction. *Applied Intelligence*, 49:206–219, 2018.

[115] R. Rudinger, P. Rastogi, F. Ferraro, and B. Van Durme. Script induction as language modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1681–1686, Lisbon, Portugal, Sept. 2015. Association for Computational Linguistics.

[116] S. Sangeetha and M. Arock. Event coreference resolution using mincut based graph clustering. In *Proceedings of the 4th International Workshop on Computer Networks & Communications*, pages 253–260, 2012.

[117] R. Schank and R. Abelson. *Scripts, plans, goals and understanding: An inquiry into human knowledge structures*. Lawrence Erlbaum Associates, 1977.

[118] A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[119] T. Shibata and S. Kurohashi. Acquiring strongly-related events using predicate-argument co-occurring statistics and case frames. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1028–1036, Chiang Mai, Thailand, November 2011. Asian Federation of Natural Language Processing.

[120] W. M. Soon, H. T. Ng, and D. C. Y. Lim. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544, 2001.

[121] M. Stede and M. Taboada. Annotation guidelines for rhetorical structure. 2017.

[122] B. M. Sundheim. Overview of the third message understanding evaluation and conference. In *Proceedings of the 3rd Conference on Message Under-*

standing, MUC3 '91, page 3–16, USA, 1991. Association for Computational Linguistics.

[123] B. M. Sundheim and N. A. Chinchor. Survey of the Message Understanding Conferences. In *Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*, 1993.

[124] J. Turian, L. Ratinov, and Y. Bengio. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, 2010.

[125] T. A. Van Dijk. *News As Discourse*. Lawrence Erlbaum Associates, 10 1988.

[126] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.

[127] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[128] M. Vilain, J. Burger, J. Aberdeen, D. Connolly, and L. Hirschman. A model-theoretic coreference scoring scheme. In *Proceedings of the Sixth Message Understanding Conference*, pages 45–52, 1995.

[129] D. Wang, P. Liu, Y. Zheng, X. Qiu, and X. Huang. Heterogeneous graph neural networks for extractive document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6209–6219, Online, July 2020. Association for Computational Linguistics.

[130] N. Weber, N. Balasubramanian, and N. Chambers. Event representations with tensor-based compositions. pages 4946–4953, 2018.

[131] Y. Wei. Document summarization method based on heterogeneous graph. In *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*, pages 1285–1289, 2012.

[132] F. Xu, H. Uszkoreit, and H. Li. Automatic event and relation detection with seeds of varying complexity. In *Proceedings of the AAAI workshop event extraction and synthesis*, pages 12–17, 2006.

[133] J. Xu and G. Durrett. Neural extractive text summarization with syntactic compression, 2019.

[134] J. Xu, Z. Gan, Y. Cheng, and J. Liu. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5021–5031, Online, July 2020. Association for Computational Linguistics.

[135] B. Yang, C. Cardie, and P. Frazier. A hierarchical distance-dependent bayesian model for event coreference resolution. *Transactions of the Association for Computational Linguistics*, 3:517–528, 2015.

[136] Z. Yang, B. Dhingra, Y. Yuan, J. Hu, W. W. Cohen, and R. Salakhutdinov. Words or Characters? Fine-grained gating for reading comprehension. In *Proceedings of the 5th International Conference on Learning Representations*, 2017.

[137] W. V. Yarlott, C. Cornelio, T. Gao, and M. Finlayson. Identifying the discourse function of news article paragraphs. In *Proceedings of the Workshop Events and Stories in the News 2018*, pages 25–33, Santa Fe, New Mexico, U.S.A, Aug. 2018. Association for Computational Linguistics.

[138] M. Yasunaga, R. Zhang, K. Meelu, A. Pareek, K. Srinivasan, and D. Radev. Graph-based neural multi-document summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 452–462, Vancouver, Canada, Aug. 2017. Association for Computational Linguistics.

[139] X. Zhang, F. Wei, and M. Zhou. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069, Florence, Italy, July 2019. Association for Computational Linguistics.

[140] Q. Zhou, N. Yang, F. Wei, S. Huang, M. Zhou, and T. Zhao. Neural document summarization by jointly learning to score and select sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 654–663, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[141] B. Zoph, D. Yuret, J. May, and K. Knight. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas, Nov. 2016. Association for Computational Linguistics.

List of Publications

[1] Y. J. Huang and S. Kurohashi. Improving shared argument identification in Japanese event knowledge acquisition. In *Proceedings of the Events and Stories in the News Workshop*, pages 21–30, Vancouver, Canada, Aug. 2017. Association for Computational Linguistics.

[2] Y. J. Huang and S. Kurohashi. Extractive summarization considering discourse and coreference relations based on heterogeneous graph. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3046–3052, Online, Apr. 2021. Association for Computational Linguistics.

[3] Y. J. Huang and S. Kurohashi. Heterogeneous graph based extractive summarization considering discourse and coreference relations. 自然言語処理, 28(2), 2021.

[4] Y. J. Huang, J. Lu, S. Kurohashi, and V. Ng. Improving event coreference resolution by learning argument compatibility from unlabeled data. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 785–795, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.