Understanding Event Structure in Text

Mohammed Aldawsari
malda021@fiu.edu

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

UNDERSTANDING EVENT STRUCTURE IN TEXT

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

by

Mohammed Aldawsari

2020

To: Dean John L. Volakis
College of Engineering and Computing

This dissertation, written by Mohammed Aldawsari, and entitled Understanding Event Structure in Text, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Ellen Thompson

Fahad Saeed

Monique S. Ross

Liting Hu

Mark Finlayson, Major Professor

Date of Defense: October 30, 2020

The dissertation of Mohammed Aldawsari is approved.

Dean John L. Volakis
College of Engineering and Computing

Andrés G. Gil
Vice President for Research and Economic Development
Dean of the University Graduate School

Florida International University, 2020

DEDICATION

Dedicated to my father.

ACKNOWLEDGMENTS

My Ph.D. journey was the period when I interacted with great individuals who have had an impact on my life. First, I am eternally grateful to my advisor, Prof. Mark Finlayson, for his constant encouragement, great advice and lots of great ideas throughout my Ph.D. study. Not only how to do research, but also I learned from him how to communicate, discuss, exchange ideas and find the positive side of negative results. This thesis would not have been possible without him.

I would like to say a big 'thank-you' to all Cognac laboratory's members for their wise advice, and constant assistance through my Ph.D. life, especially, Dr. Joshua Eisenberg, Dr. Deya Banisakher, Victor Yarlott, Labiba Jahan, Samira Zad, Mustafa Ocal, Anurag Acharya, Mireya Jurado and Adrian Perez.

I would like to thank my committee members: Professor Ellen Thompson, Professor Fahad Saeed, Professor Monique S. Ross, and Professor Liting Hu, for their valuable feedback on my research. I would also like to thank Prof. Ed Hovy and Dr. Ehsaneddin Asgari for making data available for usage in my research.

I would like to thank Prince Sattam Bin Abdulaziz University for providing me a doctoral fellowship and the Saudi Arabian Cultural Mission (SACM) for their constant support.

Finally, my family is a constant source of support. To my parents, thank you for the continuous support and love. To my brothers and sisters, thank you for always being there for me. To my wife, Maymunah and my daughters, Tala and Malath, thank you for your patience, support, and love.

ABSTRACT OF THE DISSERTATION

UNDERSTANDING EVENT STRUCTURE IN TEXT

by

Mohammed Aldawsari

Florida International University, 2020

Miami, Florida

Professor Mark Finlayson, Major Professor

Stories often appear in textual form, for example, news stories are found in the form of newspaper articles, blogs, broadcast transcripts, and so forth. These contain descriptions of current, past, or future events. Automatically extracting knowledge from these events descriptions is an important natural language processing (NLP) task, and understanding event structure aids in this knowledge extraction. Event structure is the fact that events may have relationships or internal structure, for example, they can be in a co-reference relationship with another event mention, or composed of subevents.

Understanding event structure has received less attention in NLP than is due. This work develops computational methods to automatically understand events found in narrative text and reveal their structure. In particular, I address four problems related to event structure understanding: (1) Detecting when one event is a subevent of another; (2) Identifying foreground and background events as well as the general temporal position of background events relative to the foreground period (past, present, future, and their combinations); (3) Leveraging foreground and background event knowledge to improve the extraction of event relations, specifically subevent, co-reference, and discourse-level temporal relations; and (4) Developing an event-based approach to solving the story fragment stitching problem, i.e., aligning a set of story fragments into a full, ordered, end-to-end list of story events. The latter problem is similar to the cross-document event co-reference relation task but is more challenging because the overall timeline of the story's events need to be preserved across all fragments.

For the first problem, I present a supervised machine learning model that outperforms prior models on this task and show the effectiveness of discourse and narrative features in modeling subevent relations. For the second and third problem, I demonstrate a featurized supervised model for detecting foreground and background events and illustrate the usefulness of foreground and background knowledge in event relations tasks, namely, subevent, co-reference, and discourse-level temporal relations. Lastly, I introduce a graph-based unsupervised approach and apply an adapted model merging approach to solve the story fragment stitching problem.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

An overwhelming number of stories, for example news articles and blogs, are increasingly generated daily in an unstructured form. These unstructured texts contain descriptions of current, past, or future events. An event is something that occurs (or may occur) in a certain place at a certain time (Pustejovsky et al., 2003b), described in detail in Section 1.3. An event is sometimes associated with semantic information (a.k.a., arguments) such as participants, locations, or times. If an event occurs, such as an attack, protest, natural disaster, a stock split, or companies' merger or acquisition, an overwhelming number of unstructured texts are generated about that event. This amount of unstructured text makes it arduous for humans to obtain all information related to a single event or topic. Also, manually making use of these unstructured texts becomes tedious, labor intensive, and time consuming. To overcome this difficulty, information extraction systems (IEs) have been developed to automatically extract knowledge from unstructured texts and build rich representations of its content that can be useful in various applications such as decision making support tools (Wei and Lee, 2004) and monitoring systems (Kamijo et al., 2000).

Despite the importance of automatically extracting knowledge from unstructured texts, understanding events and their relationship has received less attention in research on natural language processing (NLP). Understanding events and revealing their relations aids in understanding text and capturing all information related to events. For example, if an attack event occurred during a protest event, the attack might contain information about the protest event due to the containment relation. Also, if the temporal relations (e.g., before and after) between events are captured, then this temporal information might be helpful in various ways, e.g., useful in time-based decision systems. In general, revealing the relationships between events in the text is helpful in representing the text in many meaningful ways and useful in many downstream NLP tasks.

The current state-of-the-art systems for understanding events and their relations is still far from the desired goal. The need for methods and models that can capture the structure of events motivate me to study event structure in news narratives. Not only useful in knowledge extraction, but understanding event structure will also undoubtedly have an impact on several NLP subdomains and applications such as document summarization, storyline generation, and question answering systems.

This dissertation focuses on tackling several problems in understanding events and their structure in text. The rest of this chapter is an introduction to some terminology and definitions related to events and their relationships, as well as an overview of the dissertation.

1.2 What is a Story?

A story is defined as a sequence of events affected by characters and presented in a discourse. This is in accord with fairly standard definitions: for example, Forster (1927) said that "A story is a narrative of events arranged in their time sequence." Eisenberg and Finlayson (2017) suggested that a story is "a discourse presenting a coherent sequence of events which are causally related and purposely related, concern specific characters and times, and overall displays a level of organization beyond the commonsense coherence of the events themselves." Most importantly, these definitions point out the separation between the plot (a.k.a., the *fabula*)—which is the actual, time-ordered sequence of events—and the story itself—which is the discourse presentation of the plot.

1.3 What is an Event?

Verb meanings and their internal and temporal structure go back to the philosophical literature, precisely Aristotle's typology of events (cf. Aristotle's *Metaphysics*) (Ross et al., 1924; Ryle, 1949; Vesey, 1964). These matters find their way to the linguistic literature by way of Vendler's influential paper (Vendler, 1957). Vendler classifies verbs based on temporal properties such as termination, duration, and internal temporal structure. The four classes are states, activities, accomplishments, and achievements. States are verbs that do not have internal structure and do not change during the time which they are true (e.g., *John loves Mary*); whereas activities have internal structure and duration, but temporal termination is unnecessary (e.g., *John walked on Ocean Drive*). Accomplishments are events that have duration and temporal termination (e.g., *John consumed 3 bottles of water*); whereas achievements have an endpoint but no duration (e.g., *John arrived in Miami*).

Vendler's classes have been organized and subgrouped by various researchers including the works of Mourelatos (1978); Carlson (1981), and Bach (1986); statives and nonstatives is the most basic distinction. Bach (1986) introduced the "eventualities" term that includes all aspectual types both stative and eventive. Bach (1986) defined three classes: states, processes, and events. States are changeless (i.e., there is no perceptible change) such as *John knows Mary*, whereas processes and events differ in terms of telicity. Processes are atelic activities (i.e., have no particular end) such as *John ran*. Events are telic activities (i.e., have a culmination) such as *John builds a bookcase*.

Bach's eventuality has been adopted and covered by the term "event" in the computational semantics community (Briscoe et al., 1990; James, 1995). In the information extraction (IE) community, for example, the definition of an event is adopted, including non-verb events, and extended with more fine grained elements. Examples of prior works

Argument	Definition
Agent	Causer of the event
Patient	Affected by the event
Time	Time of the event
Location	Location of the event

Table 1.1: Definition of event argument.

in the IE community includes, but is not limited to, TimeML (Pustejovsky et al., 2003b) and the Automatic Content Extraction (ACE) program (Doddington et al., 2004).

TimeML is a markup language for events, times, and their temporal relation. An event in the TimeML schema is defined as a situation that happened or occurred as well as any predicate describing a state or circumstance in which something obtains or holds true. Similar to the TimeML schema, the ACE annotation guidelines defines an event as something that happens or occurs involving participants and can also be described as a change of state.

These two models are designed for different purposes; therefor events are tagged differently. In both schema, events can be verbs, nominalizations, adjectives, predicative clauses, or prepositional phrases and both schema consider the basic features of events (i.e., aspect, tense, modality, and polarity). TimeML tags all events, except generic events, times, and their temporal relations. In contrast, ACE only tags events from specific domain types (i.e., life, movement, transaction, business, conflict, contact, personnel, and justice) along with other information such as event type, subtype, and event participants/attributes (a.k.a., event arguments). As defined by the ACE program, an event argument is any entity or attribute that has a certain role (e.g., time and location) in the event, described in Table 1.1.

Figure 1.1 shows in bold the event of interest, and the involved arguments are underlined. Beyond the scope of this thesis but important to note is that other definitions of events in computational linguistics exist. For example, in topic detection and tracking

[*The militants*$_{agent}$] **killed** [4 women$_{patient}$] [on Friday$_{time}$] in [Kashmir$_{location}$].

Figure 1.1: Example text showing an event in bold and its arguments are underlined.

(TDT), an event can be seen as a collection of documents (e.g., news articles) that are related by a certain seminal real-world event (Allan et al., 1998).

1.3.1 Prior Work on Event Extraction

There is a voluminous amount of prior work on event extraction. Early approaches used rule-based or pattern matching algorithms for event extraction, which rely on predefined expressions or rules generated by expert knowledge (Li et al., 2002; Wei and Lee, 2004; Grishman et al., 2005). Grishman et al. (2005) presented a baseline system called Java Extraction Toolkit[1] (JET), which uses a rule-based algorithm on the training data and applies it on the test data for event extraction.

Due to the fact that rule-based approaches require expert knowledge and effort to generate rules, a substantial amount of prior work used data-driven methods by using supervised token-level classifiers powered by several token-level features such as word's lemma, surface form, and part of speech. These classifiers usually employ a pipeline approach that first determines whether or not a word is an event, following the assumption that the vast majority of events consists of a single word, and then determines the event arguments. Ahn (2006) and Hardy et al. (2006) presented logistic regression models trained to determine whether or not a word is an event and achieved a 0.60 F1 score, which is a measure for evaluating accuracy, on the ACE 2005 test set and a 0.59 F1 score on the topic of weapons of mass destruction, respectively. Subsequent works followed Ahn (2006) and Hardy et al. (2006) by using different models (e.g., support vector machines (SVMs)) or

[1] `http://cs.nyu.edu/grishman/jet/jet.html`

a different set of features (e.g., document-level features and cluster-level statistical information) (Ji and Grishman, 2008; Liao and Grishman, 2010; Hong et al., 2011). To avoid error propagation in the pipeline approach, Li et al. (2013) presented a joint model to extract events and arguments that co-occur in the same sentence using structured perceptron with beam search. Inspired by the work of Li et al. (2013), Yang and Mitchell (2016) used a similar model and a larger set of features to extract events and entities within a document context.

In recent years, neural network approaches have become popular in NLP and have shown effective results in many NLP tasks. Researchers have explored both recurrent neural networks and convolutional neural networks for event extraction (Nguyen et al., 2016; Feng et al., 2016; Liu et al., 2017; Yang et al., 2018, 2019; Tong et al., 2020). These works and specifically supervised methods suffer from the long tail issue (i.e., trigger words with frequency less than 5 account for 78.2% in the ACE 2005 corpus (Tong et al., 2020)). To overcome this issue, Tong et al. (2020) used open-domain trigger knowledge to pre-train a teacher-student model for event extraction that achieved a 78.6 F1 score. Tong et al.'s model is the state-of-the-art for event extraction on the ACE 2005 corpus.

1.4 Overview of Event Relation Tasks in NLP

In this section, I introduce an overview of relations that may exist between events. In particular, I focus on describing event co-reference resolution, subevent structure, and temporal relations. Also, I demonstrate and compare the existing corpora annotated with these relations. Even though these corpora may contain useful annotated information such as the relation between entities in text, the focus is only on events and their relations.

*John **married** Mary on May 10. **It** was a wonderful **event**.*

Figure 1.2: Example text showing in bold coreferring events.

1.4.1 Event Co-reference

Event co-reference is a linguistic phenomenon where two events are considered coreferent if both events refer to the same real-world event. That is, an event mention can be substituted/replaced by the other event and result in no semantic changes except syntax changes to preserve grammatical rules. In comparison with entity co-reference, event co-reference detection is less studied and arguably more challenging (Lu and Ng, 2018). Figure 1.2 shows an example in which bolded words (**married**, **It** and **event**) are coreferential events that refer to the same real-world event: the *marriage event*. As can be seen in the figure, all bolded words have different surface form and part of speech but are considered coreferential. There are two main tasks in event co-reference: within-document event co-reference (WDEC) and cross-document event co-reference (CDEC). The goal of the within-document task is to group coreferential events in the same document, whereas in the cross-document task the goal is to group coreferential events cross documents. Researchers have introduced several corpora annotated with event co-reference relations but with a different perspective of when to call two events coreferential. I list below some of the well-known corpora for event co-reference.

ACE 2005 The ACE 2005 corpus[2] (Walker et al., 2006) was introduced by the ACE program and consists of 599 texts with 5349 event triggers drawn from five different categories (i.e., newswire, broadcast news, broadcast conversation, weblog, and conversational telephone speech). An event in ACE 2005 is tagged with the event's arguments and several attributes (i.e., type, subtype, modality, polarity, genericity, and tense). This

[2]`https://catalog.ldc.upenn.edu/LDC2006T06`

corpus follows a strict definition that two events are coreferent if their arguments are compatible (i.e., they have exactly the same agent, patient, time, and location). Example 1.3 shows two coreferent events, but the argument compatibility constraint is violated; thus, and corresponding to the ACE 2005 annotation guideline, the two events should not be tagged as coreferent. The reason is that the patient "*John Smith and his wife*" of the "**shot**$_{e1}$" event is not compatible with the patient "*Smith*" of the "**attack**$_{e2}$" event (i.e., it only refers to a part of the "**shot**$_{e1}$" event patient argument). However, many events can be coreferent and violate the compatibility constraint, as shown in the example. Thus, this restricted constraint was one of the reasons that led to the development of other corpora such as TAC KBP corpora. Table 1.2 shows the statistics of the ACE 2005 corpus.

The killer ***shot***$_{e1}$ *John Smith and his wife* ... *The* ***attack***$_{e2}$ *against Smith.*

Figure 1.3: Example text showing two coreferring events with incompatible arguments.

# Documents	599
# Sentences	18162
# Event mentions	5349
# Event co-reference clusters	4090

Table 1.2: The ACE 2005 statistics.

TAC KBP 2017 corpus The Text Analysis Conference (TAC) Knowledge Base Population (KBP) 2017 corpus[3] is a part of the TAC KBP Event track, which was a shared task for the years of 2015, 2016, and 2017. The TAC KBP 2017 corpus is a closed-domain event corpus similar to the ACE 2005 and annotated with events and event co-reference relations as well as event arguments. Unlike ACE 2005, the TAC KBP 2017 corpus includes additional attributes and relaxed the compatibility constraint (i.e., two events are

[3]`https://tac.nist.gov//2017/KBP/Event/index.html`

coreferent even if their arguments are non-coreferential or conflicting such as *10 people killed* versus *many people killed* and *4 women on Friday* versus *4 women last week*). Moreover, the TAC KBP corpus introduced the notion of event nugget. The event nugget notion allows discontinuous words to be tagged as an event, unlike previous work where an event could be only a single word or continuous words. For instance, consider the underlined phrase in this sentence: *The crash left 40 people dead*. TimeML and ACE models tag the head word "*left*" of the phrase as the event, whereas TAC KBPs considers both words "*left*" and "*dead*". Table 1.3 shows statistics of the TAC KBP 2017 corpus.

# Documents	167
# Event mentions	4375
# Event co-reference clusters	2963

Table 1.3: The TAC KBP 2017 statistics.

ECB+ corpus The ECB+ corpus[4] (Cybulska and Vossen, 2014) is an extension to the EventCorefBank (ECB) corpus (Bejan and Harabagiu, 2010) and one of the largest datasets that includes both within and cross events and entities co-reference annotation. The ECB+ corpus consists of news articles collected from Google News and clustered into topics. Each topic consists of documents discussing the same event. The extension of ECB includes annotating more documents and increasing the level of difficulty by adding documents into a topic that discusses a different event of the same type. For example, adding documents about "*The Sudan Armed forces attack on the refugee camp*" into the topic that has documents related to "*The Israeli attack on the Fakhora school*". Table 1.4 shows the statistics of the ECB+ corpus. I used this corpus in Section 2.7.

[4]`http://www.newsreader-project.eu/results/data/the-ecb-corpus/`

# Topics	43
# Sub-topics	86
# Documents	976
# Sentences	1840
# Entity mentions	8289
# Entity co-reference clusters	2224
# Event mentions	6833
# Event co-reference clusters	2741

Table 1.4: The ECB+ statistics.

IC and HiEve corpora The Intelligence Community (IC) corpus[5] (Hovy et al., 2013) contains 100 news articles in the Violent Event domain (attacks, killings, wars, etc.). The HiEve corpus[6] (Glavaš et al., 2014) is an open domain corpus that also contains 100 news articles. Both corpora are annotated with both co-reference and subevent relations (discussed in the next Section). I used and described both corpora in details in Section 2.4.

1.4.2 Subevent Relation

Events are not atomic entities: they often have a complex internal structure that can be expressed in a variety of ways (Huttunen et al., 2002; Hovy et al., 2013). The subevent relation is one of these complex structures in which an event can be a subevent of another in discourse. An event e_j is a subevent of another event e_i if both events occur at more or less the same location and time and, more importantly, e_i acts as a collection of events, and e_j is one of them. As shown in example 1.4, the **killed**$_{e3}$ event is spatiotemporally contained by the **attacked**$_{e4}$ event; thus, we call the relation between these two events a *subevent relation*. I provide a formal definition of subevent relation in Section

[5]This corpus is not publicly available, but I obtained it via private communication with the author.

[6]`http://takelab.fer.hr/hievents.rar`

2.1. Despite the importance and usefulness of detecting subevent relations for many NLP downstream applications, there is a very little work on this task, and thus fewer annotated corpora.

To the best of my knowledge, there are only three English corpora annotated with subevent relations, namely, IC, HiEve, and Richer Event Description (RED) corpora, excluding biomedical related corpora. The IC and HiEve corpora are discussed earlier and explained in details in Section 2.4. The RED corpus consists of over 95 English newswire, discussion forum, and narrative text documents that are annotated with several event relations such as temporal, causal, and subevent relations. The RED corpus is not publicly available and only available via LDC[7].

Egyptian police have said that five protesters were ***killed***$_{e3}$ *when they were* ***attacked***$_{e4}$ *by an armed group near the Defense Ministry building in Cairo.*

Figure 1.4: Example text showing subevent relation.

1.4.3 Temporal Relation

News articles and narrative texts mostly use events to describe something that happened or will happen at a certain time. The relevant temporal information for events (e.g., the temporal order of events) is usually inferred by humans either via explicit or implicit time cues. Automatically extracting temporally relevant information is a very important aspect of many NLP downstream tasks such as question answering, information retrieval, and narrative generation and understanding.

The extraction of temporal information can be viewed as building a graph from text, where nodes are events or temporal expressions (a.k.a., timex), and edges are the tem-

[7]`https://catalog.ldc.upenn.edu/LDC2016T23`

poral relation between the nodes. Timex is a text expression that expresses the time of something that happened or how long it lasts (e.g., Tuesday and three months). Among other links, TimeML introduced a specification (TLINK) to temporally link events and temporal expression. TLINK is a specification for linking event and event (EE), event and timex (ET), and timex and timex (TT). The focus in this dissertation, specifically in Section 4.4, is on the temporal relation between events (EE).

The TimeML TLINK specification is based on Allen's interval algebra between two intervals (Allen, 1983). Figure 1.5 shows the maps between the relation types existing in Allen's interval logic and the TimeML TLINK types.

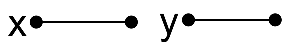
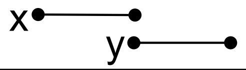
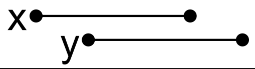
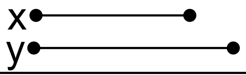
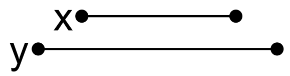
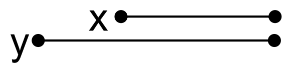
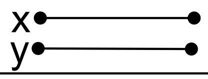
Allen's Relation	**Illustration**	**TimeML Relation**
X < Y , Y > X		X BEFORE Y , Y AFTER X
X m Y , Y m^{-1} X		X IBEFORE Y , Y IAFTER X
X o Y , Y o^{-1} X		X OVERLAPS Y
X s Y , Y s^{-1} X		X BEGINS Y , Y BEGUN_BY X
X d Y , Y d^{-1} X		X DURING Y , Y DURING_INV X (IS_INCLUDED, INCLUDES)
X f Y , Y f^{-1} X		X ENDS Y , Y ENDED_BY X
X = Y , Y = X		X SIMULTANEOUS Y

Figure 1.5: Allen's atomic relations, their illustration, and their corresponding TimeML TLINK type.

Several corpora, English and non-English, have been annotated with temporal information based on the TimeML specification such as the TimeBank 1.2 corpus (Pustejovsky et al., 2006), the TimeBank-Dense corpus (Cassidy et al., 2014) and the TDDiscourse corpus (Naik et al., 2019). The TimeBank corpus contains 183 news articles annotated with

Dataset	# documents	# Event mentions	# E-E
TimeBank	183	7935	3450
TimeBank-Dense	36	1729	8130
TDD-Man	36	1729	6150
TDD-Auto	36	1729	38302

Table 1.5: Statistics of Corpora annotated with temporal information. E-E denotes event-event temporal relations, and TDD denotes TDDiscourse corpus.

events, timex, and their temporal relations. Due to the fact that TimeBank only focuses on the most salient event in a sentence, thus annotating small portions of relations, the TimeBank-Dense corpus extended TimeBank and produced 10 times more relations per document than TimeBank (Cassidy et al., 2014). TimeBank-Dense only contains 36 documents and made a clear distinction between almost overlapped relation (e.g., *before* and *immediately before*), by reducing the relations to six, namely *before, after, includes, is-included, simultaneous*, and *vague*. The argument of reducing and making coarse-grained relations is that this fine-grained distinction (e.g., distinguishing between *before* and *immediately before*) may complicate an already difficult task, and there is no clear benefit of the fine-grained distinction yet. TDDiscourse, a recent and augmented dataset of TimeBank-Dense, focused on discourse-level temporal ordering (instead of neighboring sentences) and used the same set of temporal relations as TimeBank-Dense. I used TD-Discourse and discuss it in Section 4.4. Table 1.5 shows statistics of the aforementioned corpora.

1.4.4 Other Event Relations

Several relations can be found between events such as *causal* and *membership* relations. A causal relation is defined as when an event causes another event. For example, in this sentence *"The man was* **arrested** *because he* **killed** *his neighbor"*, the **arrested** event

The Al-Qaeda linked group which said it carried out the deadly ***attack***$_{e5}$ *against US soldiers in the Iraqi . . . The* ***operation***$_{e6}$ *is one of the heaviest* ***blows***$_{e7}$ *in the city of Mosul.*

Figure 1.6: Example text showing membership relation.

occurred because of the **killed** event, thus there is a causality relation between these two events. The membership relation holds between two events when one event is a part of the other but not necessarily at the same time and location, and both events have the same type. For instance, the **attack**$_{e5}$ and **operation**$_{e6}$ events, in Figure 1.6, are members of the **blows**$_{e7}$ event.

Unfortunately, unlike other relations, *causal* and *membership* relations have received less attention in the NLP community. Until the work of Mirza and Tonelli (2016), a few resources have been introduced for causal relations such as the PropBank (Bonial et al., 2010) and causal discourse relations in the Penn Discourse Treebank (Prasad et al., 2008). However, these resources do not cover all aspects of causality (Mirza and Tonelli, 2016). Mirza and Tonelli (2016) presented an annotation guideline for annotating causality in text and introduced the Causal-TimeBank corpus by annotating TimeBank with causal relations.

For membership relations, Hovy et al. (2013) introduced the IC corpus, which contains membership relations. Despite this useful attempt, the annotator agreement for membership relations in IC corpus is very low (0.21 Fleiss's kappa), which is not reliable given the small number of instances in the IC corpus (Hovy et al., 2013).

1.5 Overview of the Dissertation

The central goal of this dissertation is to computationally advance the understanding of events and their structure in text. To achieve this goal, I have addressed four problems:

- **Subevent Structure Detection** (Chapter §2). Despite the importance of detecting subevent relations in text, this problem has received less attention even though it is useful in knowledge extraction and many NLP tasks. This chapter advocates a novel approach to tackle this problem by leveraging discourse and narrative features. My approach outperforms prior work on this problem.
- **Foreground and Background Event Detection** (Chapter §3). Understanding the role of events and why an event is mentioned in text is a very important aspect of understanding the text as whole. One of the error sources in my subevent model (Chapter §2) is the lack of distinguishing between foreground and background events. Therefore, I introduce the task of distinguishing between foreground and background events in news articles as well as identifying the general temporal position of background events relative to the foreground. Due to the lack of corpora for this problem, I also provided a corpus annotated with foreground and background events and built a system that can distinguish between these events.
- **Integrating Foreground and Background Events into Event Relation Detection** (Chapter §4). While detecting foreground and background event properties is useful for subevent detection, it is also useful for other tasks. I demonstrate the effectiveness of foreground and background information in modeling and improving pairwise event co-reference, subevent detection, and discourse-level temporal relation extraction.
- **Event Based Fragmented Story Stitching** (Chapter §5). Finally, understanding subevents and the foreground/background event distinction are together potentially useful in new NLP tasks. In this chapter, I introduce a challenging new task, namely, stitching a fragmented story into one coherent narrative. Stories are found throughout our daily lives (e.g., news) but also a single story can be found across different media and sometimes in a fragmented way (i.e., misses/includes certain events).

This task is similar to the cross-document event co-reference but the events timeline is considered among all fragments. I provided an annotated dataset and proposed a graph-based unsupervised approach for solving this problem.

CHAPTER 2

SUBEVENT DETECTION

Recognizing the internal structure of events is a challenging language processing task of great importance for text understanding. One of the unsolved problems related to event understanding is the detection of subevents, also referred to as event hierarchy construction.

2.1 Subevent Structure

There have been efforts that have focused on detecting temporal and spatial subevent containment individually. However, it is clear that subevent detection requires both simultaneously. The subevent relationship is defined in terms of (e_i; e_j), where e_i and e_j are events: event e_j is a subevent of event e_i if e_j is spatiotemporally contained by e_i. More precisely, we say that an event e_i is a parent event of event e_j, and e_j is a child event of e_i if (1) e_i is a collector event that contains a complex sequence of activities; (2) e_j is one of these activities; and (3) e_j is spatially and temporally contained within e_i (i.e., e_j occur at the same time and same place as e_i) (Hovy et al., 2013; Glavaš and Šnajder, 2014). This subevent relationship is independent of other types of relationships, e.g., causal relationship between the events.

> *Egyptian police have said that five protesters were* ***killed***$_{e8}$ *when they were* ***attacked***$_{e9}$ *by an armed group near the Defense Ministry building in Cairo. The statement said that early this morning, the armed group* ***attacked***$_{e10}$ *the demonstrators who have for days been staging their* ***protest***$_{e11}$ *against the military government. . . . Police said that the* ***attack***$_{e12}$ *on Wednesday* ***wounded***$_{e13}$ *at least 50 protesters.*

Figure 2.1: Excerpt from the HiEve corpus (Glavaš et al., 2014). Events are in bold, and the identified events are gold annotations, but for clarity, not all annotations are included.

Figure 2.1 illustrates a text expression of a complex event hierarchy. Figure 2.2 shows a corresponding graphical representation of the hierarchy. In Figure 2.2, we see that **killed**$_{e8}$ and **wounded**$_{e13}$ are explicitly annotated as subevents of **attacked**$_{e10}$, while that event, in turn, is a subevent of **protest**$_{e11}$. Events **attacked**$_{e9}$ and **attack**$_{e12}$ are explicitly indicated as coreferent with **attacked**$_{e10}$. These relationships induce the implicit subevent relations shown by dashed lines.

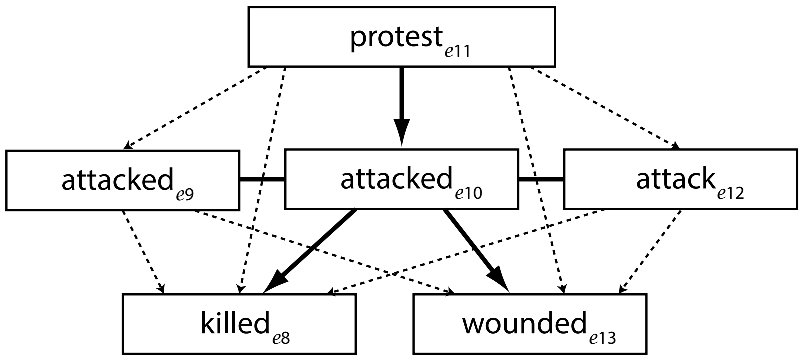


Figure 2.2: The corresponding event hierarchy of Figure 2.1. Bolded arrows indicate subevent relationships, and bolded lines indicate event co-reference relationships when they are explicitly indicated in the HiEve annotations. Dashed lines indicate an implicit subevent relationship.

2.2 Related Work

There are two pieces of prior work that are most related to my work. Araki et al. (2014) proposed a logistic regression model to classify pairs of events into four classes: co-reference, subevent, sister, and no relation. They then used sister relations and their parents to improve the system performance. Their model was trained and tested on 65 articles from the IC corpus developed by Hovy et al. (2013). Similarly, Glavaš and Šnajder (2014) used a logistic regression model to classify pairs of events into three classes: subevent relations (SuperSub and SubSuper) and no relation. They enforced structural coherence, which improved the extracted subevent relations by a 7.6% F$_1$ score. They trained and tested their approach on the HiEve corpus developed by Glavaš et al. (2014). Both approaches were evaluated using different evaluation metrics. Araki et al. evaluated their model using the BLANC evaluation metric (Recasens and Hovy, 2011) whereas Glavaš and Šnajder evaluated their model using the standard F_1 evaluation metric. Both works introduced a variety of features. The main contribution of my work is to note that the subevent detection task requires a better understanding of the discourse. Thus in this chapter, I describe a supervised machine learning approach for detecting subevent relations in text. That is, I introduce a logistic regression model using several new features, including discourse and narrative structure, and demonstrate why these features are effective in detecting subevent relations.

2.3 Features

In this section, I explain the features used in my model. As discussed in Section 1.4.2, the HiEve and IC corpora are annotated with both subevent and event co-reference relationships. I compute features over all pairs of events (e_i; e_j) where e_i precedes e_j in the text. Each pair of events is either related by a forward-pointing parent-child relationship (PC), a backward-pointing parent-child relationship (CP), or no relation (NoRel). The features can be divided into five sets, as shown in Table 2.2. In the following sections, I first illustrate the features I directly obtained from prior work (§2.3.1); next, I explain the features that were inspired by prior work but that I modified significantly (§2.3.2); and finally, I introduce my new discourse and narrative features (§2.3.3). I pre-processed texts using the `spaCy`[1] NLP tool (Honnibal and Montani, 2017). The pre-processing task is as follows:

[1]`https://spacy.io/`

- **Segmentation** The process of breaking text into sentences.
- **Tokenization** The process of breaking a sentence into lists of words and meaningful segments called tokens.
- **Lemmatization** The process of finding the lemma of a word by converting a word into its base/dictionary form, e.g., *happier* or *happiest* to *happy*.
- **Part of speech (POS)** The process of assigning the most probable major and detailed part of speech tag to each token based on the Universal POS schema[2], e.g., NOUN, VERB, and DET.
- **Syntactic dependency** The process of showing which words depend on (modify or are arguments of) other words. Figure 2.3 shows an example of the dependency relation between tokens/words in this sentence: *"The police killed the terrorist."* In Figure 2.3, the head of the sentence is the word *killed*, and both words (*police* and *terrorist*) depend on (a.k.a., are arguments of) the word *killed*, whereas the determiner *The* modifies the word *police* and the second determiner, *the*, modifies the word *terrorist*.

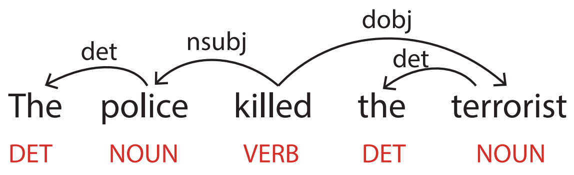


Figure 2.3: The dependency relation between words. The arc label describes the type of syntactic relation that connects a child to its head corresponding to the Universal dependency relations schema.

[2]https://universaldependencies.org/u/pos

before	when	after	later	then	until
once	during	as soon as	already	within	subsequent
since	prior to	subsequently	during and after	meanwhile	at least until
previously	prior	still	earlier	followed by	immediately
pending	while	following	at the same time	ending	shortly

Table 2.1: Temporal signals list.

2.3.1 Prior Features

I obtained most of the lexical and syntactic features, and several of the semantic features, directly from prior work on subevent detection (Araki et al., 2014; Glavaš and Šnajder, 2014). The lexical and syntactic features, as well as other features, are shown in Table 2.2. I used the `spaCy` (Honnibal and Montani, 2017) tool to compute lexical and syntactic features.

2.3.2 Modified Features

Five of my features were inspired by those in prior work, but I modified them for my system. Below I explain each feature and its representation.

Temporal Signals I observed that if a sentence mentions two events from different event hierarchies, then a temporal signal often exists between them (e.g., after and since). This is illustrated in Figure 2.4, where a temporal signal (i.e., since) exists between two events that have subevent relation. To capture this, I used Derczynski and Gaizauskas (2010) temporal signals list to find intervening temporal signal words between events and encoded this as a bag of temporal signals. Table 2.3 shows the signal list used in my experiment.

Feature Set or Feature	Representation	Description
Lexical		
Event Expression	Bag-of-Events	The surface form of e_i and e_j.
Same Lemma	Binary	Whether e_i and e_j have the same lemma.
Temporal Signals*	Bag-of-Signals	The temporal signals between e_i and e_j.
Event String Similarity	Numeric	The string similarity between surface forms of e_i and e_j using a Levenshtein measure.
Syntactic		
Major POS	One-hot	The POS of e_i and e_j (e.g., Noun and Verb).
Same Major POS	Binary	Whether e_i and e_j have the same Major POS.
POS Tag	One-hot	The POS Tag of e_i and e_j.
Same POS Tag	Binary	Whether e_i and e_j have the same POS tag.
Syntactic Dependency*	One-hot	The ancestor event of the other event in the dependency tree.
Determiner	Binary	Whether each event has a determiner.
Semantic		
Semantic Frame	Binary	Whether e_i and e_j have the same semantic frame.
Event Type*	One-hot	The event type of e_i and e_j extracted from the mapping from frames to event types.
Same Event Type	Binary	Whether event types of e_i and e_j are the same.
VerbOcean Score	Numeric	The VerbOcean score (Chklovski and Pantel, 2004) between e_i and e_j.
Semantic Similarity*	Numeric	The cosine similarity between e_i and e_j.
Ontology*	One-hot	Which event is most likely to be a parent of the other event.
WordNet Similarity	Numeric	The WordNet Similarity using (Lin, 1998; Wu and Palmer, 1994) similarity measures.
Arguments		
Co-refering Event Arguments*	One-hot	Whether the arguments of e_i and e_j corefer.
# of Coreferring Args	Numeric	The number of coreferring arguments between e_i and e_j.
Event in the Other's Args	One-hot	Whether one event is mentioned in one of the other event's arguments.
Discourse & Narrative		
Sentence Distance	Numeric	The number of sentences between e_i and e_j.
Event Distance	Numeric	The number of events between e_i and e_j.
Same Sentence	Binary	Whether e_i and e_j are in the same sentence.
Reported Speech	Binary	Whether an event mention is mentioned in a direct speech.
Non Major Mention	Binary	Whether the sentences, in which the events are mentioned, share co-referential non major mentions (see Section 2.3.3).
RST-DTs Relation	One-hot	The discourse relation between elementary discourse units (EDUs) (see Section 2.3.3).

Table 2.2: Features used in my model. Novel features are underlined. Features modified from prior work are marked with an asterisk.

Over 90 Palestinians and one Israeli soldier have been ***killed***$_{e14}$ <u>*since*</u> *Israel* ***launched***$_{e15}$ *a massive air . . .*

Figure 2.4: An example text showing the temporal signal since between two events that have a subevent relation.

Syntactic Dependency Both prior systems encoded a feature that captured whether one event in a pair was an immediate child (i.e., governed) of the other. I expand that to checking for ancestry more generally. This is encoded as a one-hot vector.

Event Type I identified the event type of each event corresponding to the 33 ACE 2005 event types shown in Table 2.3. Technically, Liu et al. (2016) identified a possible mapping from frames to the 33 ACE event types. FrameNet (Fillmore et al., 2003) is a lexical resource of manually identified semantic frames. FrameNet has over 1000 different frames, and each frame consists of lemmas with POS that can evoke the frame. For example, both *strike.NOUN* and *bomb.VERB* lemmas belong to the *Attack* frame. I used the `SEMAFOR` tool (Das et al., 2010) to extract the event frame and used Liu et al.'s mapping to identify the event type of each event. This feature is encoded as a one-hot vector.

be-born	marry	divorce	injure	die
transport	transfer-ownership	transfer-money	start-org	merge-org
declare-bankruptcy	end-org	attack	demonstrate	meet
phone-write	start-position	end-position	nominate	elect
arrest-jail	release-parole	trial-hearing	charge-indict	sue
convict	sentence	fine	execute	extradite
acquit	appeal	pardon		

Table 2.3: The 33 ACE 2005 event types.

Semantic Similarity A popular idea in NLP is representing words by vectors (a.k.a., Word2Vec or word embeddings). Word2Vec is a two-layer neural model that processes text to capture the semantic meaning of words by grouping similar words together in vector space. Technically, the Word2Vec model takes as input a large corpus of text and outputs a set of vectors that represent words in that corpus. These vectors can then be used as features in many NLP downstream tasks. `FastText`[3] (Mikolov et al., 2018) is one of the tools that can be used to train a Word2Vec model on a large text. The intuition behind using the word embeddings as a feature is that if there is a subevent relation between two events then their semantics might be similar; thus, their corresponding vectors are close to each other in the vector space. Therefore, to capture the semantic similarity between a pair of events, I used the cosine similarity measure between the pairs vectors. I used one of the `FastText` pre-trained models (i.e., wiki-news-300d-1M) to compute events vectors. This feature is encoded as a numeric feature.

Most Likely Parent Event For this feature and similar to Araki et al. (2014), I count the number of times in the training data that a particular event lemma and POS pair is observed as a parent of another event lemma/POS pair. For a pair (e_i and e_j), if the lemma and POS of e_i is more often found as a parent of e_j, this is encoded as the vector (1,0,0); if the opposite is true, this is encoded as (0,1,0). If there were no observations, this is encoded as (0,0,1). Prior work did not take into account the part of speech or the direction of the subevent relationship.

Co-referring Event Arguments To extract event arguments, I used two models, namely, `Allennlp`[4] semantic role labeling (SRL) (Gardner et al., 2018; He et al., 2017) for verbs, and the `SEMAFOR` tool (Das et al., 2010) for non-verb events. Allennlp's SRL is trained and evaluated on OntoNotes5.0 (Weischedel et al., 2011), whereas `SEMAFOR` is trained and evaluated on the FrameNet corpus (Fillmore et al., 2003). OntoNotes5.0 is training data for training semantic role labeling systems (i.e., answering the question of *who did*

[3]`https://fasttext.cc/docs/en/english-vectors.html`

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

Label	Description	Label	Description
ARG0	Agent, operator	ARG1	Thing operated
ARG2	Explicit patient	ARG3	Explicit argument
ARG4	Explicit instrument	ARGM-LOC	Locative
ARGM-TMP	Temporal	ARGM-MNR	Manner
ARGM-DIR	Direction	ARGM-DIS	Discourse
ARGM-EXT	Extent	ARGM-PRP	Purpose
ARGM-NEG	Negation	ARGM-MOD	Modal
ARGM-REC	Reciprocals	ARGM-PRD	Secondary Predication
ARGM	Bare ArgM	ARGM-ADV	Adverbials

Table 2.4: Argument labels and their descriptions.

what to whom?). The OntoNotes5.0 dataset is annotated with verbal propositions and their arguments. Each verb (a.k.a., predicate) is marked with several possible arguments, defined in Table 2.4. Text in Figure 2.5 shows a predicate, *build*, and its arguments. Unlike OntoNotes5.0, which only focuses on verbs, a predicate in FrameNet can be a verb, noun, adjective, or adverb. Each frame in FrameNet is associated with a set of roles called *frame elements* such as Agent, Patient, and Time. In this feature, for measuring arguments' co-reference between a pair of events, I only considered *ARG0, ARG1, ARGM-TMP* and *ARGM-LOC* from the `Allennlp` SRL model and *Agent, Patient, Time* and *Location* from `SEMAFOR` model. When measuring arguments' co-reference, I allowed ARG0/Agent to match ARG0/Agent or ARG1/Patient and vice versa, and I also examined LOC/Location and ARGM-TMP/Time modifying arguments. I used the `Allennlp` co-reference model (Lee et al., 2017) to resolve the arguments. This feature is encoded as a six-place binary vector.

However, voters decided that [if the stadium was such a good idea $_{ARGM-ADV}$*] [someone* $_{ARG0}$*] [would* $_{ARGM-MOD}$*]* **[build** $_{predicate}$**]** *[it* $_{ARG1}$*] [himself* $_{ARGM-REC}$*], and rejected it 59% to 41%.*

Figure 2.5: An example text showing a predicate and its arguments. Other predicates such as decided and rejected are not considered in this example for clarification.

2.3.3 New Features

The new features are divided into three types: two discourse features, one narrative feature, and two semantic features.

- **Discourse Features** I investigate the importance of discourse features for detecting subevents. I introduced two new features: rhetorical structure and reported speech.

  1. **Rhetorical Structure** Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) is a hierarchical model that aims to identify the discourse structure of a text. The text is first segmented into Elementary Discourse Units (EDUs) which in turn are linked in binary or multi-way discourse relations (see Carlson and Marcu, 2001). Rhetorical analysis is beneficial in many NLP tasks, including sentiment analysis (Somasundaran, 2010; Lazaridou et al., 2013; Bhatia et al., 2015), text generation (Prasad et al., 2005), information extraction (Maslennikov and Chua, 2007), question answering (Verberne et al., 2007) and co-reference resolution (Cristea et al., 1998; Joty et al., 2015). Therefore I hypothesized that discourse structure could be useful to the subevent detection task. For example, Figure 2.6 shows the discourse relation of type *explanation* between two EDUs in which a pair of events are found and have a subevent relation. I employ the `CODRA`[5] discourse parser (COmplete probabilisticDiscriminative framework for performing Rhetorical Analysis; Joty et al., 2015) to build a discourse tree of each text. I use Neumann (2015)'s implementation[6] for post-processing the `CODRA` output to build a graph representing the result. I then extract the rhetorical relation between event mentions using

[5] `http://alt.qcri.org/tools/discourse-parser/`

[6] `https://github.com/arne-cl/discoursegraphs`

the rhetorical relation between the EDUs in which the events are found. The feature is encoded as a one-hot vector covering all 16 main relation classes.

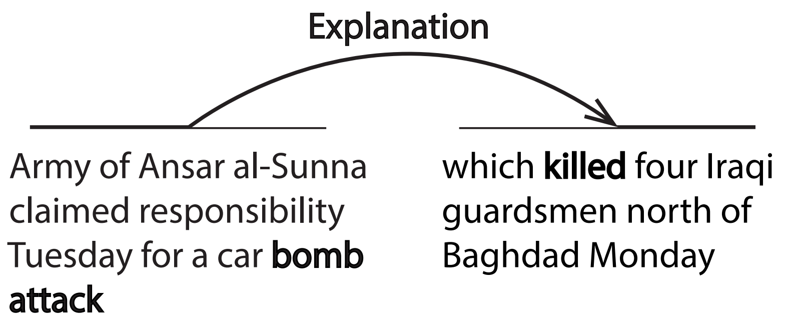

Explanation
Army of Ansar al-Sunna claimed responsibility Tuesday for a car **bomb attack**
which **killed** four Iraqi guardsmen north of Baghdad Monday

Figure 2.6: An example showing the discourse relation between two EDUs in which the target events are found.

Consider Figure 2.7. When applied to this text, the discourse parser identifies the relation between **raid**$_{e18}$ and **killed**$_{e19}$ as an *Elaboration* relation. Furthermore, the parser also captures a *Topic-Change* relation between **offensive**$_{e21}$ and each of **killed**$_{e16}$, **wounded**$_{e17}$, **raid**$_{e18}$, **killed**$_{e19}$, and **injured**$_{e20}$.

*One Palestinian was **killed**$_{e16}$ and at least four others were **wounded**$_{e17}$ in an Israeli air **raid**$_{e18}$ near the southern Gaza town of Rafah on Sunday, <u>Palestinian security sources</u> said. … <u>Palestinian security sources</u> said that one Palestinian bystander was **killed**$_{e19}$ and at least four others were **injured**$_{e20}$. … Israeli troops continued a massive ground and air **offensive**$_{e21}$ in the Gaza Strip on Sunday.*

Figure 2.7: Excerpt from the IC corpus. Events relevant to explaining the discourse features are bolded. Mentions relevant to explaining the narrative feature are underlined. Note that, for clarity, not all events marked in the corpus are bolded here (e.g., Reporting events such as said).

Although the discourse parser is useful primarily for providing information about inter-sentential relationships between events, it can also give useful information about intra-sentential relationships. Consider Figure 2.8. For this text, the discourse parser finds the *Background* relation between **abduction**$_{e22}$ and each of **killed**$_{e23}$ and **rescued**$_{e24}$.

Mahsud, a former prisoner at Guantanamo Bay, is being hunted for the ***abduction***$_{e22}$ *of two Chinese engineers, which ended last Thursday when commandos* ***killed***$_{e23}$ *five kidnappers and* ***rescued***$_{e24}$ *one Chinese.*

Figure 2.8: A sentence where intra-sentential discourse relations are useful for discovering subevent relations.

2. **Reported Speech** I also observed that subevents are often reported in direct and indirect speech. Direct speech is speech set off with quotes, while indirect speech is speech reported without quotes. I only considered direct speech in this work, primarily because it is easy to detect; however, subevents are also likely to be reported in indirect speech as can be seen in Figure 2.7 where **killed**$_{e19}$, and **injured**$_{e20}$ (which are subevents of **raid**$_{e18}$) are mentioned in indirect speech.

- **Narrative Feature**

1. **Non-Major Mentions** Similar to the event co-reference resolution task, the entity co-reference resolution task aims to group or cluster expressions in text that refer to the same entity. Consider the text in Figure 2.9. The entity co-reference task is to build a system that can identify that *Mahsud, Mahsud, Mahsud, his, his, Mahsud, his, he, he, and He* mentions, shown in red in the figure, refer to the same entity and both *The military* and *the military* mentions, shown in blue in the figure, refer to the same entity. The entity co-reference resolution has been a very critical component in many NLP downstream tasks such as information extraction and named entity linking (Durrett and Klein, 2014; Ji et al., 2014).

   I introduced what I am calling a *narrative* feature that I found informative in detecting subevent relations. This feature recognizes that other entities mentioned in a sentence in addition to those in the event arguments can be useful in subevent detection. This feature is narrative in the sense that it takes into account whether an entity is central to the story in the text.

In particular, I observed that many sentences that share an event hierarchy also share some coreferring non-major mentions in addition to event arguments. Despite this, certain entities are so central to the text that they are mentioned nearly everywhere and are thus not especially informative, e.g., the mention *Mahsud* and its referring expressions that are shown in red in Figure 2.9.

> *On Tuesday militants believed to be Mahsud loyalists attacked an army convoy, killing five soldiers and wounding seven. Initial reports put the toll at three soldiers dead and five injured. Mahsud, a former prisoner at Guantanamo Bay, is being hunted for the abduction of two Chinese engineers, which ended last Thursday when commandos killed five kidnappers and rescued one Chinese. The military has launched several previous operations in the wild South Waziristan region against hundreds of militants, including foreigners, who are believed to be hiding there with local help. Mahsud became commander of Al-Qaeda-linked militants after the military killed his predecessor Nek Mohammad in a missile strike in June on his hideout near Wana, the main town in the district. Mahsud returned to his rugged homeland after he was released in March from the US Guantanamo Bay detention center in Cuba. Pakistan authorities have said he was not on the list of Pakistanis released from the center and might have returned via Afghanistan. He was captured there in late 2001 after the US-led invasion ousted the Taliban regime.*

Figure 2.9: A news article showing one of the major mentions in red and one of the non-major mentions in blue.

Therefore I filter out these *major mentions* and encode as a binary feature whether or not the sentences that contain the pair of interest share a non-major mention such as *the military* mention, shown in blue in Figure 2.9.

The trick, of course, is defining what is a *major mention*. A simple and effective way of filtering out major mentions is to measure the distribution of co-reference chain lengths (normalized to the number of the corresponding article's chains), and discard all chains with a length above a certain threshold. This threshold can be tuned to the data. In my experiment, I estimated the mean and standard deviation of the distribution of co-reference chains in each text and filtered out chains that were longer than a single standard deviation above the mean. In Figure 2.9, the threshold of the corresponding article is 2, thus *the military*, which is mentioned only twice, is not considered a major mention. Also, Figure 2.7 shows two sentences that share a non-major mentions (i.e., Palestinian security sources) and two events that have a subevent relation (**raid**$_{e18}$ and **killed**$_{e19}$).

- **Semantic Features**

  1. **Event in the Other's Arguments** I observed that if an event hierarchy is expressed within a sentence, one of the events is often mentioned as part of the other event's arguments, as can be seen in Figure 2.10, where the **attack**$_{e25}$ event appears as ARG0 of **killed**$_{e26}$. Although this feature is related to the *Syntactic Dependency* feature, an event's arguments are not always syntactically dependent on the event head, so it adds useful information. I also include the number of coreferring event arguments as a numeric feature.

The Al-Qaeda linked Army of Ansar al-Sunna claimed responsibility on Tuesday for a car bomb ***attack***$_{e25}$ *which* ***killed***$_{e26}$ *four Iraqi guardsmen.*

Figure 2.10: A sentence where one event appears inside the argument for another event.

2.4 Corpora

I used two corpora, the IC corpus (Hovy et al., 2013) and the HiEve corpus (Glavaš et al., 2014) to train and test my model. The IC corpus contains 100 news articles in the Violent

	IC	**HiEve**
# of sentences	1,973	1,377
# of tokens	48,737	34,917
# PC relations, original	472	609
# PC relations, transitive closure	1632	1802
# CP relations, original	257	351
# CP relations, transitive closure	1665	1846
# NoRel relations	48567	42094
Avg # of sents. per article	19.7	13.7
Avg # of sents. in an event boundary	6.2	8.3
Avg # of events per article	30.5	26.0
Avg # of events in each hierarchy	5.2	7.0
Avg # of hierarchies per article	3.29	2.19

Table 2.5: Statistics of the IC and HiEve corpora.

Event domain (*attacks, killings, wars, etc.*). The HiEve corpus is an open domain corpus that also contains 100 news articles. Both corpora are annotated with both co-reference and subevent relations. The inter-annotator agreement for the IC corpus is 0.467 Fleiss's kappa for subevent relations. The approach proposed for temporal relations by UzZaman and Allen (2011) was used to measure the inter-annotator agreement in HiEve, resulting in 0.69 F_1. There is a small conceptual difference between the annotation of subevent relations in the two corpora. The annotation of subevents in the IC corpus follows Hovy et al. (2013), where they argued that the event identity can be divided into three categories: *fully identical*, *quasi-identical* (a.k.a., partial co-reference), and *fully independent* (not identical). Quasi-identity, in turn, appears in two ways: *membership* or *subevent*. As mentioned in Section 1.4.4, *membership* is defined as when an event is a set of multiple instances of the same type of event, and the other event is one of the instances. In contrast, the HiEve corpus considers the *membership* relation as a subevent relation. When training on the IC corpus, I considered only the subevent relations, and ignored the membership relations.

For both corpora, I extend the annotations by computing the transitive closure of both co-reference and subevent relations according to the following rules in Figure 2.11, where e_i, e_j, and e_k are event mentions, $\equiv$ indicates event co-reference, $e_i > e_j$ indicates e_i is a parent of e_j , and $e_i < e_j$ indicates e_i is a child of e_j. All of these rules are taken from work by Glavaš et al. (2014). I confirmed that this closure produces a consistent graph, and thus is insensitive to the order of computation of the closure. Table 2.5 shows the statistics of both corpora.

1. $(e_i \equiv e_j)$ & $(e_j \equiv e_k) \Rightarrow (e_i \equiv e_k)$
2. $(e_i > e_j)$ & $(e_j > e_k) \Rightarrow (e_i > e_k)$
3. $(e_i < e_j)$ & $(e_j < e_k) \Rightarrow (e_i < e_k)$
4. $(e_i > e_j)$ & $(e_j \equiv e_k) \Rightarrow (e_i > e_k)$
5. $(e_i > e_j)$ & $(e_i \equiv e_k) \Rightarrow (e_k > e_j)$
6. $(e_i < e_j)$ & $(e_j \equiv e_k) \Rightarrow (e_i < e_k)$
7. $(e_i < e_j)$ & $(e_i \equiv e_k) \Rightarrow (e_k < e_j)$

Figure 2.11: The transitive closure rules.

2.5 Experiment

In this section, I describe the experiment and the evaluation metrics used to measure the performance of my model. Then I compare the performance of my model with previous models, specifically those of Araki et al. (2014) and Glavaš and Šnajder (2014).

	IC corpus			HiEve corpus		
	Training	**Test**	**Total**	**Training**	**Test**	**Total**
# articles	80	20	100	80	20	100
# PC (avg.)	1299.2	332.8	1632	1484	318	1802
# CP (avg.)	1317.8	347.2	1665	1456.4	389.6	1846
# NoRel (avg.)	39469	9098	48567	35621.2	6472.8	42094

Table 2.6: Average statistics of the folds. PC stands for parent-child relation. CP stands for child-parent relation. NoRel stands for no relation.

2.5.1 Experimental Setup

I use the Linear SVM classifier from `scikit-learn` library[7] for classification over the gold annotated event mentions. Linear SVM can handle multi-class classification using a one-vs-rest scheme (Pedregosa et al., 2011). Most of the parameters are default parameters[8], but to address the issue of the data imbalance as shown in Table 2.6, I use the parameter `class_weight=balanced` to assign a higher misclassification penalty on the minority class (PC and CP). I conducted 5-fold cross-validation for the experiment. Average fold statistics are shown in Table 2.6.

2.5.2 Evaluation and Result

I use the same evaluation metrics used in previous models. Araki et al. (2014) evaluated their model using the BLANC evaluation metric (Recasens and Hovy, 2011), whereas Glavaš and Šnajder (2014) evaluated their model using the standard F_1 evaluation metric. The BLANC metric computes the F_1 score for two separate links (i.e., positive (pos) and negative (neg) links). When applying BLANC to the system output of the three classes,

[7]`https://scikit-learn.org/stable/`

[8]`penalty=l2,C=0.01, random_state=0, max_iter=1000, class_weight=balanced, multi_class=ovr.`

the 3x3 confusion matrix is converted to a 2x2 confusion matrix as a binary decision of the system to each class. Precision and recall of positive and negative links are computed as follows:

$$Precision_{pos} = \frac{tp}{tp + fp}$$
$$Recall_{pos} = \frac{tp}{tp + fn}$$

$$Precision_{neg} = \frac{tn}{tn + fn}$$
$$Recall_{neg} = \frac{tn}{tn + fp}$$

and the BLANC metric is computed as follow:

$$BLANC = \frac{F_{1pos} + F_{1neg}}{2}$$

$$where\ F_{1pos} = \frac{Precision_{pos} * Recall_{pos}}{Precision_{pos} + Recall_{pos}}$$

$$and\ F_{1neg} = \frac{Precision_{neg} * Recall_{neg}}{Precision_{neg} + Recall_{neg}}$$

The results of the performance averaged across all five folds on the three classes (PC, CP, and NoRel) are shown in Table 2.7 using both evaluation metrics on both corpora. Table 2.8 shows the comparison between my model and previous models. Although it is not clear to me how Araki et al. handled the direction of the subevent relation, I take the average of my model classes (PC and CP) and compare it with the subevent class in Araki et al.'s work. For Glavaš and Šnajder (2014), I consider only their *coherent* model, which is the best model that does not use the gold co-reference relations. Therefore, in Table 2.8, the reported result of all models is the average of both classes (PC and CP). As can be

		Evaluation Metrics							
		F_1 Score			BLANC				
					Pos Links		Neg Links		Avg
Corpus	Relation	P	R	F_1	P	R	P	R	F_1
HiEve	PC	0.576	0.807	0.67	0.661	0.832	0.989	0.973	0.857
	CP	0.661	0.832	0.733	0.576	0.807	0.990	0.971	0.825
	NoRel	0.98	0.945	0.962	0.980	0.945	0.625	0.830	0.836
IC	PC	0.469	0.564	0.506	0.455	0.549	0.982	0.973	0.735
	CP	0.454	0.550	0.492	0.468	0.564	0.983	0.975	0.743
	NoRel	0.966	0.905	0.958	0.966	0.949	0.461	0.557	0.729

Table 2.7: My model result on the IC corpus and HiEve corpus using BLANC and F_1 standard evaluation metrics. PC stands for parent-child relation. CP stands for child-parent relation.

shown in Table 2.8, my model outperforms both prior models by 15 and 5 percentage points. Also, the table shows that the precision is lower than the recall, which indicates that the subevent detection task is still a difficult and complex task that needs more work.

2.5.3 Discussion

As shown in Table 2.7, my model performs worse on the IC corpus than on HiEve. This is not surprising given the large difference in annotation agreement between IC and HiEve as well as the removal of *membership* relations in the IC corpus (see Section 2.4). In addition to its lower annotation agreement, the IC corpus is also domain-specific, with events only related to the intelligence community. This makes general resources and tools (e.g., VerbOcean, WordNet) less effective.

I investigated the importance of each of the five feature sets (Table 2.2) to my model by retraining it while leaving out one set at a time. In order of importance, they are (1) Syntactic, (2) Semantic, (3) Discourse & Narrative, (4) Lexical, and (5) Arguments. The importance of the syntactic features derived from the fact that children events are most often mentioned in the same sentence as their parent events. The three most important

		F_1 Score			BLANC				
					Pos Links		Neg Links		Avg
Corpus	Model	P	R	F_1	P	R	P	R	F_1
IC	Araki et al. (2014)	-	-	-	0.144	0.333	0.993	0.981	0.594
	Araki et al. Re-Impl.	0.242	0.285	0.262	-	-	-	-	-
	My model	0.461	0.557	**0.499**	0.461	0.557	0.983	0.974	**0.739**
HiEve	Glavaš and Šnajder (2014)	0.766	0.565	0.65	-	-	-	-	-
	Glavaš and Šnajder Re-Impl.	-	-	-	0.562	0.750	0.983	0.971	0.813
	My model	0.618	0.82	**0.701**	0.618	0.82	0.99	0.972	**0.841**

Table 2.8: My model performance compared to previous models. Each row represents the average of both classes parent-child (PC) and child-parent (CP). Because the prior systems did not report both metrics, I approximated the metrics for those systems by reimplementing them.

features among the Semantic features are *Most Likely Parent Event*, *Event Type*, and *Semantic Frame*. For the Lexical feature set, the *Event Feature* and *Temporal Signals* are the most important.

2.6 Error Analysis

Inspection of the results revealed several types of errors, aside from the usual noise introduced by the various sub-components, such as the discourse parser or co-reference systems. I cluster the errors into three types: (1) an event pair that should be classified as *PC* but is classified as *CP* and vice versa (about 28%); (2) an event pair wrongly classified as NoRel (missed subevent relation; about 12%); (3) an event pair that is actually NoRel that is wrongly classified as subevent (PC or CP; about 60% of the errors).

Type 1: PC as CP or vice versa About a third of the model errors were this type. Most of the errors are a result of an incorrect *Event Type* feature. This feature plays a major role in capturing the direction of the subevent relation. For example, if an event e_i

with event type *Die* occurs in the text before an event e_j with event type *Attack*, then the direction of the relation is mostly *child-parent* relation. But if e_j occurs before e_i, then the direction of the relation is mostly *parent-child*. If the event type is unknown for one of the event mentions, my model commonly fails to capture the direction.

Type 2: Incorrect NoRel Most of the type 2 errors occur when an event is far away from its related event in terms of the number of intervening sentences. The larger the distance between events, the more likely that the model makes this error. For this type of error, I calculated the average number of sentences and the average number of events intervening between a missed pair of events, for which the model should capture its subevent relation, and found that when the distance is greater than 9 sentences and the number of events is greater than 14, it is more likely that the model would conduct this error. Subevents tend to be close to their parents in the text, as shown in Table 2.5. Moreover, I observed that the *Non-Major Mention* and *Discourse Relation* features (described in Section 2.3.3) were less useful the larger the distance between the events.

Type 3: False Positive PC or CP Most of the errors were of this type. There were a variety of causes, but the most common was when a sentence contained multiple event hierarchies. Consider Figure 2.12 in which the sentence contains two different event hierarchies, namely, one hierarchy containing **offensive**$_{e29}$ and another containing **abduction**$_{e30}$.

Over 90 Palestinians and one Israeli soldier have been ***killed***$_{e27}$ *since Israel* ***launched***$_{e28}$ *a massive air and ground* ***offensive***$_{e29}$ *into the Gaza Strip on June 28, three days after the* ***abduction***$_{e30}$ *of one Israeli soldier by Palestinian militants in a cross-border* ***raid***$_{e31}$.

Figure 2.12: Excerpt from the IC corpus (Hovy et al., 2013) showing a passage that results in an error of Type 3.

In Figure 2.12, **killed**$_{e27}$ and **launched**$_{e28}$ are subevents of **offensive**$_{e29}$, whereas **abduction**$_{e30}$ is a subevent of **raid**$_{e31}$. When processing this example, the discourse parser failed to capture the discourse relation between **offensive**$_{e29}$ and **abduction**$_{e30}$ because both events are in the same EDU. Moreover, even though I introduced features such as temporal signals (*after*, *since*, etc.) to capture the subevent relation between intra-sentential events, this error can still occur if the intra-sentential events are syntactically related (i.e., **killed**$_{e27}$ syntactically dominates **abduction**$_{e30}$, or there is a causal relation between events).

Based on this observation, I ran an experiment on the IC corpus to examine the impact on subevent detection of having two different events in the same sentence. I constructed a subset of the IC corpus (58 articles), which excluded all articles that contain at least one sentence with two different event hierarchies, and re-ran my main experiment. Under these conditions, the model performance increased by 6 and 4.6 points F_1 on *PC* and *CP* classes, respectively (because of the smaller set, I used 3 folds instead of 5). Returning to the original corpus, I observed that two different event hierarchies are mostly found in compound and complex sentences, and one of them is usually a background event. This observation indicates that splitting compound or complex sentences into two simple sentences in advance might be useful in detecting subevents. Even though the discourse parser does this splitting automatically, this split is not currently propagated to the other features.

2.7 Improving Event Co-reference using Subevent Relation

In this section, I investigate the importance of detecting subevent relations for event co-reference improvement. The event co-reference task aims to identify clusters of coreferring events that exist in a document. The goal of this experiment is to measure the performance of an event co-reference classifier before and after including the subevent relation as a feature. For this experiment, I used the ECB+ corpus, described in Section

	Train	Validation	Test	Total
# of documents	527	190	198	915
# of coreferring pairs	877	314	545	1736
# of non-coreferring pairs	14544	4371	10206	29121
# of event mentions	3659	1239	1772	6670
Related topics	1–36	2,5,12,18,21,23,34,35	36–46	

Table 2.9: The ECB+ WDEC statistics. The number of documents does not match the number of documents mentioned in Table 1.4 because some of the documents do not have within-document co-reference chains.

1.4.1, which is annotated with both within and cross-document event co-reference. I only consider the task of within-document event co-reference (WDEC) because my subevent detection model is limited to detecting within-document subevent relations. The statistic of the annotation of WDEC relations in the ECB+ corpus is shown in Table 2.9.

2.7.1 Related Work

Different assumptions and definitions of event co-reference have led to the creation of several corpora, e.g., ACE 2005, TAC KBP, IC and ECB+, explained in Section 1.4.1. Many researchers have worked and conducted experiments on ACE 2005 (Chen and Ji, 2009; Chen and Ng, 2015), TAC KBP (Lu et al., 2016; Peng et al., 2016; Lu and Ng, 2017), IC (Hovy et al., 2013; Liu et al., 2014), and ECB+ (Lee et al., 2012; Cybulska and Vossen, 2014; Kenyon-Dean et al., 2018). In my experiment, I compared my model to the recent work of WDEC (Kenyon-Dean et al., 2018) reported on the ECB+ corpus. Kenyon-Dean et al. (2018) is the most recent work on the ECB+ corpus for detecting WDEC relations. Kenyon-Dean et al. (2018) introduced a neural network-based model that clusters event mentions based on the vector representation of the event mention and its context, e.g., all of the five tokens following the event mention. Kenyon-Dean et al. (2018) trained and evaluated their model on the ECB+ corpus's split, shown in Table 2.9.

2.7.2 Model and Features

I built a pairwise logistic regression model that detects whether a pair of events (e_i and e_j) corefer. I used `scikit-learn` implementation of logistic regression with parameters[9] chosen by conducting a grid search technique using `scikit-learn's` *GridSearchCV*[10] method over the training and validation sets combined. Over the result of the pairwise model on the test set, I employed a transitive closure approach to group all events that belong to the same cluster. That is, for each document's pairs, the classifier confidence scores are sorted (from highest to lowest), and the transitivity rule is applied over all the document's pairs (i.e., if e_i corefers with e_j and e_j corefers with e_k, then e_i corefers with e_k). Following previous work on ECB+ (Kenyon-Dean et al., 2018), I trained the classifier on the ECB+ training and validation sets, topics are shown in Table 2.9, using the following features:

- **Lexical and Syntactic** I used a binary feature to determine whether e_i and e_j share the same lemma. Syntactically, I used three features, namely, the major POS, POS tag, and the dependency relation between the target pairs (e_i and e_j). Each feature is encoded as a one-hot vector, a process of converting categorical variable into a numeric vector that machine learning algorithms can understand. I used the `spaCy` tool (Honnibal and Montani, 2017) to compute both lexical and syntactic features.
- **Event as an Entity** Some nominal events can be resolved by any of the available entity co-reference resolution systems; thus, I used a binary feature to indicate whether the two events e_i and e_j corefer using the `Allennlp` neural model (Lee et al., 2017).

[9]penalty=l2,C=0.0001,solver=liblinear,multi_class=ovr,random_state=0,class_weight=balanced, max_iter=2000

[10]cv=3, scoring=f1_macro

- **Semantic Similarity** I calculated the cosine similarity between e_i and e_j embeddings using the `FastText` (Mikolov et al., 2018) pre-trained model (wiki-news-300d-1M). This feature is encoded as a numeric feature.
- **Verb Class** VerbNet (VN) (Kipper et al., 2008) is the largest online verb lexicon that organizes verbs in classes based on Levin's verb classification [1993] that groups verbs according to shared syntactic behaviors. VerbNet extended Levin's classes and organized verbs into hierarchical classes where each class is described by frames, thematic roles, and selectional restrictions on the arguments, ensuring that members of a class are syntactically and semantically coherent. For example, in VN, *eat, chew, gobble*, and *devour* belong to the class *verbs of ingesting*. I used a binary feature to indicate whether the two events e_i and e_j belong to the same VN class. For non-verb events, I employed a simple heuristic approach to convert a non-verb event to a verb event based on the concept of the derivationally related form in WordNet, a database of English words linked by semantic relations including synonyms, hyponyms, and meronyms (Miller, 1995). The concept of derivationally related forms is defined in WordNet as those terms with different POS but have the same root form and that are semantically related (Miller, 1995). Algorithm 1 shows the approach of converting a non-verb event into a verb event using the NLTK WordNet Interface (Bird and Loper, 2004). Note that if the algorithm returns *None*, then the assumption is that the two events e_i and e_j do not belong to the same VN class.
- **Co-referring Event Arguments** I used a binary feature to indicate whether any of the arguments (i.e., ARG0, ARG1, TMP, and LOC) of the two events e_i and e_j corefer. I used the `Allennlp` SRL model (Gardner et al., 2018; He et al., 2017) to extract arguments and the `Allennlp` co-reference model (Lee et al., 2017) to resolve the arguments.

Algorithm 1: Algorithm for converting a non-verb event to a verb event

```
Input: Non-verb event
Output: Verb event
1  Function convertToVerbEvent(event):
       // synsets() function from NLTK WordNet Interface
2      sysnsets ← synsets(event, pos = event.pos)
3      if ¬ sysnsets is empty then
4          relatedVerbs ← empty list
5          foreach $synset \in sysnsets$ do
               // lemmas() function from NLTK WordNet Interface
6              foreach $lemma \in synset.lemmas()$ do
7                  if $lemma.pos == event.pos$ then
                       // $derivationally\_related\_forms()$ function from NLTK
                          WordNet Interface
8                      foreach $form \in lemma.derivationally\_related\_forms()$ do
9                          if $form.pos == VERB$ then
10                             $relatedVerbs.add(form)$
11                         end
12                     end
13                 end
14             end
15         end
16         if ¬ relatedVerbs is empty then
17             return $findMostFrequentVerb(relatedVerbs)$
18         end
19         return $None$
20     end
21     return $None$
22 End Function
```

- **Subevent Relation** I used my subevent model prediction, described in Section 2.1, on the ECB+ corpus as feature.Specifically, I used the type of the subevent relation (i.e., parent-child, child-parent, or no-relation) between the two events e_i and e_j as a feature.

2.7.3 Evaluation and Result

For training and testing my model on the ECB+ corpus, I followed Kenyon-Dean et al. (2018) setup split shown in Table 2.9. For evaluation, I used the official CoNLL scorer[11] (Pradhan et al., 2014) and report the average of the five standard metrics, namely, **MUC** (Vilain et al., 1995), **B**^{3} (Bagga and Baldwin, 1998), **CEAF-e** (Luo, 2005), **CEAF-m** (Luo, 2005) and **BLANC** (Luo et al., 2014). Below I describe each score. Let *K* be the set of gold clusters, and *R* be the set of system clusters.

MUC The recall is computed based on the minimum number of links that need to be added to the system clusters to obtain the gold clusters. Precision is switching the role of gold and system clusters. The recall is defined as follows:

$$\frac{\sum\left(|k_i| - p\left(k_i\right)\right)}{\sum\left(|k_i|\right) - 1} \tag{2.1}$$

where $k_i \in K$ and $p(k_i)$ is the set of partitions that is generated by intersecting k_i with the corresponding system cluster.

B^3 This score is a mention-based metric that computes the overall precision and recall based on each individual mention's recall and precision. The recall for each mention is computed as the fraction of the correct mentions that are included in the system cluster. The Recall is defined as follows:

$$\sum\sum\frac{|k_i \cap r_j|^2}{|k_i|} \tag{2.2}$$

where $k_i \in K$ and $r_j \in R$. Similar to **MUC**, precision is switching the role of gold and system clusters.

[11] `https://conll.github.io/reference-coreference-scorers/`

CEAF-e and CEAF-m The assumption of the CEAF metric is that each gold cluster should only be mapped to one system cluster, and vice versa. It uses a similarity measure (i.e., the Kuhn-Munkres algorithm) to compute an optimal alignment (g^*) between the gold clusters and the system clusters. The alignment can be defined as a one-to-one mapping function g whose score $\Phi(g)$ is defined as follows:

$$\Phi(g) = \sum \theta(k_i, g(k_i)) \tag{2.3}$$

where $k_i \in K$ and θ is a function that computes the similarity between a gold cluster and a system cluster. Given the optimal alignment (g^*), whose Φ value is the largest among all possible alignments, the recall and the precision of CEAF are computed as follows:

$$Recall = \frac{\Phi(g^*)}{\sum_{i=1}^{|k|} \theta(k_i, k_i)} \tag{2.4}$$

$$Precision = \frac{\Phi(g^*)}{\sum_{j=1}^{|R|} \theta(r_j, r_j)} \tag{2.5}$$

Luo (2005) defines two similarity functions θ_3 and θ_4 that result in mention-based CEAF-m and entity-based CEAF-e, respectively.

$$\theta_3(k_i, r_i) = |k_i \cap r_i| \tag{2.6}$$

$$\theta_4(k_i, r_i) = \frac{2\,|k_i \cap r_i|}{|k_i| + |r_i|} \tag{2.7}$$

There is no agreement in the literature on the best evaluation metric for event co-reference tasks. Thus and for a fair comparison with the state-of-the-art model (Kenyon-Dean et al., 2018), I reported the aforementioned evaluation metrics and the mean of MUC, B^3, and CEAF-e (a.k.a., CoNLL). Even though the goal of this experiment is to measure the performance of the classifier with and without the subevent relation feature,

Model	MUC R	MUC P	MUC F	B^3 R	B^3 P	B^3 F	CM F	CE R	CE P	CE F	BLANC F	CoNLL F	PW F
KD2018	57	69	63	90	94	92	86	90	86	88	75	81	-
My model	62	83	71	91	97	94	89	94	88	91	79	85	53
My model+	63	86	73	91	97	94	90	94	88	91	80	**86**	**54**

Table 2.10: My model's performance compared to the state-of-the-art model (Kenyon-Dean et al., 2018) on ECB+ WDEC task. KD2018 denotes Kenyon-Dean et al.'s work. PW stands for pairwise.

my model outperforms the state-of-the-art model on the ECB+ WDEC task, shown in Table 2.10. It is clear that the model performance increases slightly when I include the subevent relation feature. However, it has been shown in the literature that the transitivity rule propagates errors when clustering corefering pairs based on the classifier pairwise decision. Therefore and to accurately measure the impact of the subevent relation feature, I included the classifier performance on the pairwise decision using the standard F_1 score, shown in Table 2.10. As result, it is clear from the table that the subevent relation feature has a slight impact on the classifier performance by increasing the performance by 1 point.

CHAPTER 3

FOREGROUND AND BACKGROUND EVENT DETECTION

One of the error sources in my subevent model reported in Chapter 2 is the lack of distinguishing between foreground and background events. Therefore, I hypothesize that determining whether an event in a news article is a foreground or background event would be useful not only in subevent detection, but also in other event relations such as temporal relation extraction and event co-reference detection. In this chapter, I introduce the task of distinguishing between foreground and background events in news articles as well as identifying the general temporal position of background events relative to the foreground period (past, present, future, and their combinations). Identifying the general temporal position is a coarser analog to detailed, pairwise temporal relation extraction, and provides an intermediate step to ease the integration of discourse information into temporal understanding of the text. Chapter 4 shows the importance of using foreground and background knowledge in modeling event relations.

3.1 Definition

Grimes et al. (1975) defined foreground events as the events that form the skeleton of a story, whereas background events add supporting information. Following Grimes et al. (1975), I define *foreground* events as those that comprise the main topic of a news article, as indicated by the headline. In contrast, *background* events add supporting or contextual information. Figure 3.1 shows a snippet of text with foreground events in red and background events in other colors, divided into six general temporal position categories, as illustrated in Figure 3.2 and defined in Table 3.1. Note that while the document creation time (DCT) usually occurs after the foreground period, there is no reason why the DCT could not appear within or before it.

A ***car bomb***$_{e32}$ ***damaged***$_{e33}$ *half a city block in Istanbul Tuesday while the Prime Minister* ***attended***$_{e34}$ *a peace* ***conference***$_{e35}$*, which is scheduled from Monday to Wednesday. No* ***casualties***$_{e36}$ *were* ***reported***$_{e37}$*. The terrorist group behind the* ***attack***$_{e38}$ *has been* ***on the run***$_{e39}$ *from the military since the first major* ***bombing***$_{e40}$ *in 1998. The group* ***promised***$_{e41}$ *more* ***bombings***$_{e42}$ *soon, while the military* ***said***$_{e43}$ *that special security measures have been* ***implemented***$_{e44}$ *and would remain in place for the foreseeable future.*

Figure 3.1: An example text with foreground events marked in red, and background events in other colors, as defined in Figure 3.2.

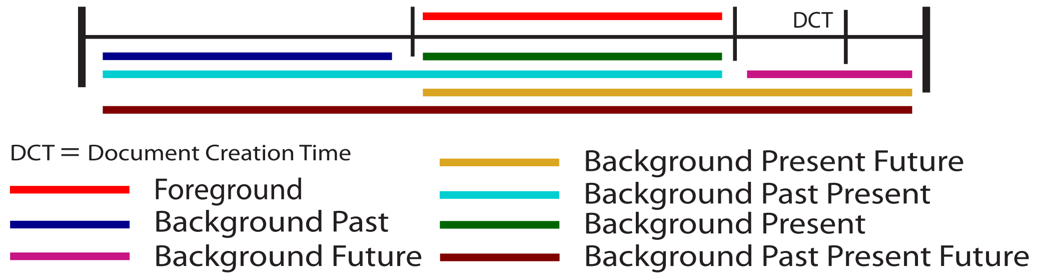


Figure 3.2: An illustration of the relative temporal position of foreground events in relation to background event categories. The document creation time (DCT) is assumed to occur after the foreground events, but this is not strictly necessary.

Background Past (BPast) events end before the foreground events begin.
Background Past Present (BPastPres) events start before and continues during the foreground period.
Background Present (BPres) events happen within the foreground event period.
Background Present Future (BPresFut) events begin during the foreground period and continue in the future.
Background Future (BFut) events begin after the foreground event period.
Background Past Present Future (BAll) events begin in the past, continue during the foreground period, and into the future.

Table 3.1: Background Event Categories, which are distinguished by their temporal position relative to the foreground period.

3.2 Task

The ability to automatically extract foreground and background events could guide document understanding and potentially be helpful in many natural language processing tasks

> *Unidentified gunmen have* ***shot dead***$_{e45}$ *the eldest son of Yemeni Transport Minister Ahmed Mussaed Hussein in the capital Sanaa, police sources in Yemen* ***told***$_{e46}$ *AFP on Friday. . . . Ahmed Mussaed Hussein was* ***named***$_{e47}$ *transport minister in 1994 after the* ***civil war***$_{e48}$.

Figure 3.3: Excerpt from the IC corpus (Glavaš et al., 2014) showing Foreground, Background, and Other classes. The named$_{e47}$ and civil war$_{e48}$ events are annotated as Background events, told$_{e46}$ as an Other event, whereas shot dead$_{e45}$ as a Foreground event.

such as temporal relation extraction (Naik et al., 2019), summarization (Zhang et al., 2018), and storyline generation (Zhou et al., 2018). The task is distinguishing between foreground and background events, as well as identifying the general temporal position of background events relative to the foreground period. More precisely, the task is to classify an event as *Foreground*, *Background*, or *Other*, and additionally assign background events to one of the six possible general temporal positions relative to the foreground period, defined in Table 3.1 and illustrated in Figure 3.2. I assume events are provided through some other process. The *Other* category includes events that are neither foreground nor background, such as generics or reporting events (e.g., *reported*$_{e37}$ in Figure 3.1). For example, in Figure 3.3, the headline of the news article, from which the excerpt is extracted, is "*Yemeni minister's son assassinated*" (i.e., this article was written to report the assassination of the Yemeni minister's son). Given this example, a model should classify **shot dead**$_{e45}$ as a *Foreground* event, **told**$_{e46}$ as an *Other* event, and both **named**$_{e47}$ and **civil war**$_{e48}$ as *Background* events (more precisely *BPast*). Both events are classified as *Background* events because they add some background information of certain entities (e.g., the Yemeni transport minister) that is not the reason this article was written. Additionally, both events are classified as *BPast* because they ended before the foreground events began.

3.3 Prior Work

Both Upadhyay et al. (2016) and Choubey et al. (2018) demonstrated approaches for identifying the central event in news articles. Upadhyay et al. (2016) proposed a rule-based system to identify the central event in a human-generated document summary. They evaluated their system on human-generated summaries from the New York Times Corpus (Sandhaus, 2008), where the central event had been identified. Similarly, Choubey et al. (2018) used several rule-based systems and statistical classifiers to identify the most important event in a news article. They trained and evaluated their systems on 30 news articles from the RED corpus (Mitamura et al., 2015) and 74 news articles from the KBP 2015 corpus (O'Gorman et al., 2016). Both were focused only on identifying a single central event, whereas I seek to label all events in a document as either *Foreground*, *Background*, or *Other*.

Huang et al. (2016) demonstrated an approach to placing events in news articles into three coarse temporal categories: *Past* events that have already occurred; *On-Going* events that are currently happening; and *Future* events that may happen. In that work, the temporal category was relative to the document creation time (DCT) and did not distinguish between foreground and background events. In contrast, my work seeks to mark the general temporal position of all background events relative to the foreground period.

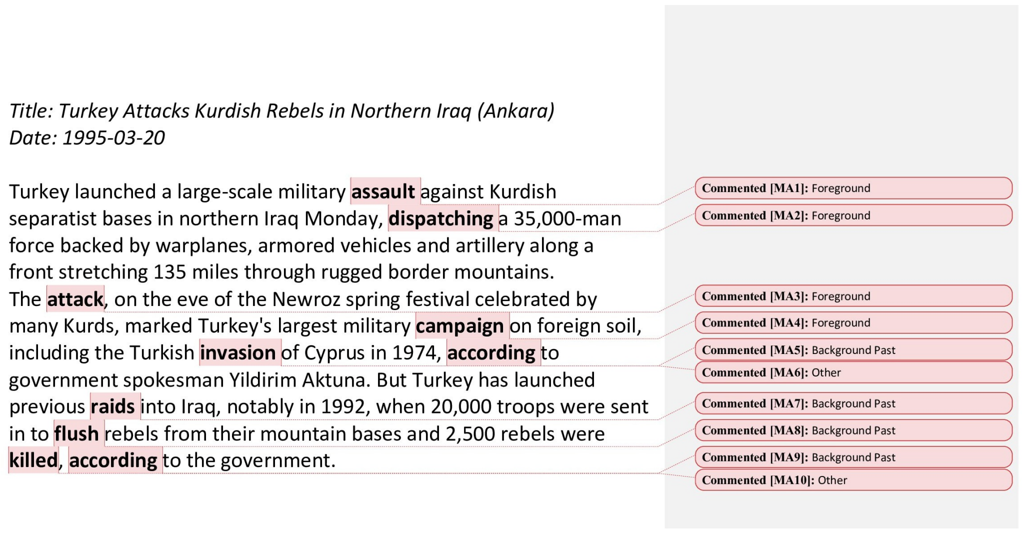
3.4 Corpus

Due to the lack of annotated corpora, two annotators[1] and I worked on the annotation of foreground and background events. We annotated 99 news articles from the Intelligence

[1]two members of the Cognac laboratory (Deya Banisakher, Ph.D., and Adrian Perez, undergraduate student)

Community (IC) corpus (Hovy et al., 2013). The IC corpus contains 100 news articles, but one article was merely a list of events rather than being a narrative. We used the gold event mentions that had been annotated on the corpus. The definition of *event* in Hovy et al. follows that of TimeML (Pustejovsky et al., 2003a; Sauri et al., 2006), which has been well studied and shown to be reliably annotatable:

> We mean both events and states when we say 'event'. A *state* refers to a fixed, or regularly changing, configuration of entities in the world, such as 'it is hot' or 'he is running'. An *event* occurs when there is a change of state in the world, such as 'he stops running' or 'the plane took off'. (Hovy et al., 2013, p. 21)



Title: Turkey Attacks Kurdish Rebels in Northern Iraq (Ankara)
Date: 1995-03-20

Turkey launched a large-scale military **assault** against Kurdish separatist bases in northern Iraq Monday, **dispatching** a 35,000-man force backed by warplanes, armored vehicles and artillery along a front stretching 135 miles through rugged border mountains. The **attack**, on the eve of the Newroz spring festival celebrated by many Kurds, marked Turkey's largest military **campaign** on foreign soil, including the Turkish **invasion** of Cyprus in 1974, **according** to government spokesman Yildirim Aktuna. But Turkey has launched previous **raids** into Iraq, notably in 1992, when 20,000 troops were sent in to **flush** rebels from their mountain bases and 2,500 rebels were **killed**, **according** to the government.

Commented [MA1]: Foreground
Commented [MA2]: Foreground
Commented [MA3]: Foreground
Commented [MA4]: Foreground
Commented [MA5]: Background Past
Commented [MA6]: Other
Commented [MA7]: Background Past
Commented [MA8]: Background Past
Commented [MA9]: Background Past
Commented [MA10]: Other

Figure 3.4: A working example of the annotation of foreground and background events including the temporal position of background events.

3.4.1 Annotation Process

Two of the annotators labeled each event in the IC corpus with one of eight categories: *Foreground*, *Other*, or six varieties of *Background* (listed in Table 3.1). The annotators were educated and given several working examples (one of these examples is shown in Figure 3.4) with the following definitions:

- **Background Past (BPast)**: Event that is not central to the topic of the news article and has started and ended before the foreground events. Below (1 and 2) are examples of the background past category, and the target events are underlined:

  1. "*Somalia has been without a government since the* *overthrow* (**BPast**) *of dictator Mohamed Siad Barre in January 1991.*" The title of the article where this example is found is "*Heavy fighting reported in breakaway Somaliland.*"
  2. "*But Turkey has launched previous* *raids* (**BPast**) *into Iraq, notably in 1992, when 20,000 troops were sent in to* *flush* (**BPast**) *rebels from their mountain bases and 2,500 rebels were* *killed* (**BPast**)*.*" The title of the article where this example is found is "*Turkey Attacks Kurdish Rebels in Northern Iraq (Ankara).*"

- **Background Past Present (BPastPres)**: Event that is not central to the topic and had started before foreground events but is still ongoing (i.e., it still overlaps with foreground events). Below (1 and 2) are examples of the background past present category.

  1. "*Barzani, leader of a feudal family which has* *fought* (**BPastPresent**) *Baghdad for decades.*" The title of the article where this example is found is "*Bloodshed shows up failure of Kurdish self-rule by Patrick Rahir.*"
  2. "*the two Iraqi groups that have set up a de facto government in northern Iraq under cover of Provide Comfort, have been fighting (BPastPresent) intermittently since April.*" The title of the article where this example is found is "*Turkey Attacks Kurdish Rebels in Northern Iraq (Ankara).*"

- **Background Present (BPres)**: Event that is not central to the topic and overlaps with foreground events, and there is no indication that the event started before the foreground events. Below is an examples of the background present category.

    1. "*Al-Alaami is an offshoot of Harkatul Mujahideen , which is* <u>*battling*</u> *(***BPresent***) Indian rule in the divided Himalayan region of Kashmir.*" The title of the article where this example is found is "*Pakistani militant held over Musharraf death plot.*"

- **Background Present Future (BPresFut)**: Event that begins during the foreground period and continues in the future. Event **implemented**$_{e44}$, in Figure 3.1, is an example of background present future.

- **Background Future (BFut)**: Event that begins after the foreground event period. Event **bombings**$_{e42}$, in Figure 3.1, is an example of background future.

- **Background Past Present Future (BAll)**: Event that begins in the past, continues during the foreground period, and into the future. Event **conference**$_{e35}$, in Figure 3.1, is an example of background past present future.

- **Foreground (F)**: Event that is central to the topic that prompted the author to write the news article. Below (1 and 2) are examples of foreground events.

    1. "*One Palestinian was* <u>*killed*</u> (**F**) *and at east four Others were* <u>*wounded*</u> (**F**) *in an Israeli air* <u>*raid*</u> *(***F***).*" Note that *killed*, *wounded*, and *raid* are all foreground events because they are all central to the topic of the article even though *killed* and *wounded* are in past tense.

    2. "*The metro workers'* <u>*strike*</u> (**F**) *in Bucharest has entered the fifth day.*"

- **Other (O)**: The other category includes any other events not belonging to any of the aforementioned categories (e.g., Other may include, generic (1), hypothetical (2), or reporting events (3).

    1. "*Jews are prohibited from killing (***O***) one another.*" Or "*Protests (***O***) are often facilitated by...*"
    2. "*If the Israelis strike (***O***), the US will surely be dragged (***O***) into a larger conflict.*"
    3. "*The operation will be of limited duration and the forces involved will be withdrawn immediately following the elimination of the targets, a government statement said (***O***).*"

The annotators were provided with some guidelines. Below is some of this information:

- Given a news article text (xml file from the IC corpus) along with a CSV file containing the targeted events of the article, write next to each event its corresponding category (Foreground, Background, or Other) and additionally assign the background events to one of the six possible general temporal positions relative to the foreground period, defined in Table 3.1.
- Always remember the reason why this article was written to distinguish between foreground and background events in general.
- In case of ambiguity, make use of the previous annotation of event co-reference and subevents in the IC corpus, i.e., if you cannot determine the temporal position of an event, refer to its coreferring events (if any) to better understand the event.
- Take advantage of the previously annotated subevent relations because parent-child events always have the same main category (Foreground and Background), but not

always the same temporal position (i.e., BPast, BPastPres, BPres, BPresFut, BFut, and BAll).

Disagreement between the two annotators was adjudicated by the third annotator. The overall agreement was 0.69 Cohen's κ (Landis and Koch, 1977). Cohen's Kappa metric is used to calculate the agreement between annotators. The value of Cohen's Kappa metric has a range of -1 to 1, where 1 denotes the ideal value of Cohen's Kappa. Table 3.2 shows the value of Cohen's Kappa and the corresponding agreement.

Cohen's Kappa value	agreement
Below 0	no agreement
Between 0 and 0.2	slight agreement
Between 0.2 and 0.4	fair agreement
Between 0.4 and 0.6	moderate agreement
Between 0.6 and 0.8	substantial agreement
Between 0.8 and 0.1	almost perfect agreement

Table 3.2: The value of Cohen's Kappa (Landis and Koch, 1977).

Table 3.3 shows agreements for individual classes as well as the statistics of the corpus. Note that in the corpus *BAll* only occurred 5 times, and *BPresFut* not at all. Table 3.3 shows the characteristics of the corpus and label counts.

3.5 Model

I built a featurized logistic regression classifier powered by several features divided into five categories: Lexical, Syntactic, Semantic, Discourse, and Time. I used a logistic regression classifier from the `scikit-learn` library (Pedregosa et al., 2011) for classification over the gold annotated event mentions. The classifier handles multi-class classification using a one-vs-rest scheme. Most of the parameters were left at their default set-

Articles	99	
Sentences	1,955	
Tokens	48,737	
Event Mentions	4,086	
Avg. Sentences / article	19.7	
Avg. Tokens / article	487.4	
Avg. Events / article	30	κ
Foreground	1,501	0.66
Background Past (BPast)	851	0.66
Background Past-Present (BPastPres)	365	0.61
Background Present (BPres)	89	0.21
Background Present-Future (BPresFut)	0	-
Background Future (BFut)	160	0.43
Background Past-Present-Future (BAll)	5	0.66
Other	1,115	0.90
Overall Markings / Agreement	4,086	0.69

Table 3.3: Corpus Statistics.

tings [2]. I addressed data imbalance (shown in Table 3.3) by using the `class_weight=balanced` parameter to assign a higher mis-classification penalty to the minority class. I conducted 5-fold cross-validation for the experiment.

3.6 Features

Lexical and Syntactic Temporal signals (e.g., after and before) often occur before background events. I used the temporal signals list collected by Derczynski and Gaizauskas (2010). This feature is encoded as a bag of signals capturing whether a temporal signal is present in the text between the target event and the immediately preceding event. For syntactic features, I use the major part of speech (POS), tense, and aspect, all encoded as

[2] `penalty=l2,C=0.1,random_state=42, max_iter=1000 class_weight=balanced, solver=liblinear, multi_class=ovr.`

a one-hot vector. I used `spaCy` (Honnibal and Montani, 2017) to compute both lexical and syntactic features.

Semantic BERT (Bidirectional Encoder Representations from Transformers) is a deep learning model (Devlin et al., 2018) that has given state-of-the-art results on a verity of NLP downstream tasks. BERT is a multi-layer bidirectional transformer trained on plain text for masked word prediction (i.e., predicting the next word given a sequence of words) and next sentence prediction tasks (i.e., whether the second sentence is the actual next sentence of the first sentence when given a pair of sentences). The BERT model can be fine-tuned for NLP downstream tasks or used as a fixed feature extractor (i.e., getting an encoded representation of a sentence). In my experiment, I computed an event contextualized representation using (Akbik et al., 2018)'s implementation of BERT model *bert-base-uncased* (Devlin et al., 2019) to capture the semantics of an event. The vector for an event is defined as the weighted sum of all subword embeddings extracted from BERT's last layer. I also captured the semantic frame of the event using the SEMAFOR tool (Das et al., 2010), encoded as a one-hot vector.

Discourse I employed two discourse features: RST discourse relation and the position of the event's sentence in the text. As mentioned earlier in Section 2.3.3, Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) is useful for many NLP tasks, including sentiment analysis (Bhatia et al., 2015) and information extraction (Maslennikov and Chua, 2007). I used the `Feng-Hirst` discourse parser (Feng and Hirst, 2014) to build a discourse tree of each text, and post-processed the output to build a graph using Neumann (2015) library. For events after the first, I extracted the rhetorical relation between the target event mention and the immediately preceding event. This feature is encoded as a one-hot vector covering all 16 main relation classes. We also captured the

	Fine-grained				Coarse-grained			
Model	**Class**	**Prec.**	**Rec.**	F_1	**Class**	**Prec.**	**Rec.**	F_1
My Model	Foreground	0.75	0.71	0.73	Foreground	0.73	0.74	0.73
	BPast	0.66	0.64	0.65	Background	0.72	0.72	0.72
	BPastPres	0.52	0.62	0.56				
	BPres	0.19	0.18	0.19				
	BFuture	0.34	0.46	0.39				
	Other	0.94	0.92	0.93	Other	0.94	0.92	0.93
	macro$_{avg}$	0.57	0.59	0.57	**macro**$_{avg}$	0.80	0.79	0.79
	micro$_{avg}$	0.72	0.72	**0.72**	**micro**$_{avg}$	0.78	0.78	**0.78**
Baseline (MFC)	**macro**$_{avg}$	0.17	0.06	0.09	**macro**$_{avg}$	0.33	0.12	0.18
	micro$_{avg}$	0.37	0.37	0.37	**micro**$_{avg}$	0.37	0.37	0.37
Baseline (Coref)	**macro**$_{avg}$	0.21	0.14	0.15	**macro**$_{avg}$	0.42	0.34	0.35
	micro$_{avg}$	0.34	0.34	0.34	**micro**$_{avg}$	0.46	0.46	0.46

Table 3.4: My model's performance on all classes. Background is abbreviated as (B).

position of the event's sentence in the discourse. This was encoded as a real number, normalized to a value between 0 and 1 by the number of sentences in the article.

Time I compute the difference, in days, between the date of the event mention and the date in the first sentence. If there is no date in the first sentence, I use the document creation time. The date of the event mention is taken to be any date used as an argument to the event, or otherwise the nearest date that appears in the sentence; if the event has neither, I assume the difference is zero. I normalized both dates to a calendar value using the `HeidelTime` utility (Strötgen and Gertz, 2013). The difference is then encoded as a one-hot vector feature with three possible values: negative, zero, or positive.

3.7 Experiment

I discuss in this section the design of the experiment (§3.7.1), baselines (§3.7.2), the performance of my model (§3.7.3), and the importance of the four feature sets (§3.7.4).

3.7.1 Experiment Setup

I conducted two experiments: in the first (the *fine-grained* condition), I used all classes from Table 3.3 except for two (*BAll* and *BPresFut*). Note that I merged the *BAll* class with the *BPres* class due to the small number of examples, and *BPresFut* had no examples in the corpus. In the second experiment (the *coarse-grained* condition) I collapsed all background classes into one.

3.7.2 Baselines

In addition to a most frequent class (MFC) baseline, I designed a strong baseline inspired by the observation that the central event of a document usually has many co-referential event mentions (Choubey et al., 2018). This baseline operates as follows: (1) Mark an event as *Foreground* if it is part of an event co-reference chain and the length of that chain is longer than or equal to the average of the lengths of event co-reference chains for each article (the event co-reference chains are identified based on the IC gold annotation); (2) Mark an event as *Other* if it is a reporting event corresponding to the IC gold annotation; (3) Otherwise, mark the event as *BPast* for the fine-grained condition (the most frequent Background class), or *Background* for the coarse-grained condition.

3.7.3 Evaluation and Result

I evaluated my model performance using both the macro and micro F_1 metric. Table 3.4 shows my model's performance under both conditions (the *fine-grained* condition and the *coarse-grained* condition). As shown in bold in Table 3.4, my model outperforms all baselines under both conditions. It achieves a reasonable performance of %0.72 F_1 micro and %0.57 F_1 macro under the *fine-grained* condition and %0.78 F_1 micro and %0.79 F_1 macro under the *coarse-grained* condition.

3.7.4 Discussion

As shown in Table 3.4 the model performance in the fine-grained condition is lower compared to the coarse-grained performance, which is not surprising given the increased number of classes (and thus reduced data) and general difficulty of detecting temporal relationships.

I investigated the importance of each of the four feature sets to my model under the fine-grained condition by retraining while leaving out one set at a time. In order of importance, they are semantic (35% performance loss), discourse (4%), time (2%),and syntactic and lexical (2%). Apparently, the most important feature set is semantic features. The BERT vector is the most important feature for all classes, but the frame feature contributed more to the *Other* class. This is because most of the events in this class are reporting events and were captured by the *Statement* frame. In the discourse set, the event's sentence position and discourse relation contributed equally to the model. The time feature contributed most to the *BPast* class because the *BPast* events were mostly associated with past temporal dates. The syntactic and lexical features were the least contributing features to the model. I realized that when I dropped the contextualized embedding, the syntactic features contributed the most to the model. By replacing BERT embeddings with ELMo (Peters et al., 2018) and Fasttext (Bojanowski et al., 2017) embeddings under the fine-grained condition, the performance decreases by 4% and 13%, respectively. Therefore, in this setting, I hypothesize that the syntactic features were mostly captured by the contextualized embeddings.

3.8 Error Analysis

Upon detailed inspection, I was able to discern several error classes aside from the usual noise introduced by the various sub-components. I observe that the model wrongly classi-

	Foreground	BPast	BPastPresent	BPresent	BFuture	Other
Foreground	-	22%	44%	5%	20%	9%
BPast	59%	-	22%	9%	4%	6%
BPastPresent	48%	32%	-	4%	12%	4%
BPresent	60%	21%	9%	-	4%	6%
BFuture	53%	15%	22%	6%	-	4%
Other	31%	24%	18%	2%	25%	-

Table 3.5: Fine-grained labeling error percentage between actual labels (rows) and predicted labels (columns).

fies Foreground events as Background if the event appears towards the end of the article. In the analysis, I also observe that this mislabeling occurs when the event is referred to in conjunction with some sort of temporal reference. For example, in an article regarding the capture of two people, in the sentence, *"The captured bomb-maker, Sami Muhammad Ali Said al-Jaaf, was* **seized**$_{e49}$ *in Baghdad on Jan. 15"*, the word **seized**$_{e49}$ is labeled as a Background event even though it is directly tied to the foreground. As shown in Table 3.5, this mislabeling constitutes 91% of the foreground event labeling error. Similarly, Background events that appear early in the article are often mistaken for Foreground events. My model mistaking Background events as Foreground comprises 91% of the model's background labeling error. The model also wrongly classifies foreground events as *Other* (9% of the foreground labeling error) if the event mention looks like a reporting event due to the missing sense (e.g., *claimed* is used in the construction *claimed lives*, but can be mistaken for a reporting event). Another common error was the lack of explicit discourse or temporal information (e.g., a date) for identifying background events.

Within the fine-grained labeling of background events, I see that errors occur mainly in making the distinction between events that are *BPast* and *BPastPresent*. Of the fine-grained error, the mislabeling of *BPast* as *BPastPresent* was 22% of the error (see Table 3.5); the labeling of *BPastPresent* as *BPast* constituted 32%. This being the largest error in the sub-classification task makes sense given that the two classes are quite similar.

Consider the the text shown in Figure 3.5. In this example, the **operations**$_{e50}$ event is a *BPast* event incorrectly labeled as *BPastPresent*. I believe this is due to the model not being sensitive to the precursory descriptors like the word *suspended*. For without that descriptor it would imply that the **operations**$_{e50}$ event is still ongoing.

Israeli security forces have been on high alert to guard against possible terror attacks by Hamas, which has suspended ***operations***$_{e50}$ *against Israel since its spiritual leader and founder Sheikh Ahmed Yassin was released from Israeli jail last October.*

Figure 3.5: Text showing an event that is subject to the common mislabeling of BPastPresent for BPast.

With regard to general temporal position, changes of tense related to the document creation time (i.e., an event is in the past relative to the DCT, but in the future relative to the foreground period) caused difficulties in distinguishing between *BFut* and *BPast*. Though this error did not occur frequently, the failure to distinguish between the two classes comprises 19% of the overall fine-grained Background error, as shown in Table 3.5. I see the model struggle with examples such as *"Another cell was uncovered* ***last fall***$_{e51}$ *, when the police carried out an operation against a group of Algerian and Moroccan radicals who were believed to be* ***planning***$_{e52}$ *an attack on Madrid 's High Court and perhaps other targets*." Difficulties in distinguishing between background classes, in general, were often the result of the writer assuming some commonsense or world knowledge on the part of the reader to infer the temporal relationship.

CHAPTER 4

INTEGRATING FOREGROUND AND BACKGROUND EVENTS INTO EVENT RELATION DETECTION

This chapter aims to investigate the impact of integrating foreground and background knowledge into three NLP tasks. The goal is to demonstrate the importance of using foreground and background knowledge as features in modeling event relations, namely, subevent detection, event co-reference resolution, and temporal relation extraction.

4.1 Introduction

To validate the importance of capturing background and foreground events as well as the temporal position of background events relative to the foreground events, I experimented with incorporating this feature into three different NLP tasks, namely, subevent detection, event co-reference resolution, and temporal relation extraction. The goal of this experiment is to measure the performance with and without including Foreground/Background fine-grained classes as features. Even though some of the experiments I developed along the way outperform the state of the art, the emphasis here is on the contribution of these features in these tasks. All experiments were performed under the *fine-grained* condition, demonstrated in Section 3.7.1.

4.2 Subevent Detection Task

As explained in Section 2.1, the subevent detection is the process of identifying when one event is a subevent of another. That is, a pair of events is classified into one of the three classes: *parent-child, child-parent, or no relation (NoRel)*, corresponding to the direction in the discourse flow. For the subevent experiment, I used my model demonstrated in Section 2.1, which is the state of the art in modeling subevent relation. In addition to the features explained in Section 2.3, I integrated the gold annotation of background and foreground events as features into the model and re-trained it with the new features. That is, for each event of a pair of events, I included the Foreground/Background fine-grained features as a one-hot vector.

Model	**Prec.**	**Rec.**	**F$_1$**
Subevent Model (Section 2.1)	0.45	0.56	0.50
+Fine-grained Labels	0.50	0.61	**0.55**

Table 4.1: Subevent experiment result.

Using the same evaluation metrics BLANC and F$_1$, Table 4.1 shows an increase in the performance by 5% after including the Foreground/Background fine-grained features. This shows that these features are beneficial in modeling subevent relations. I performed an analysis of the increase and found that these features helped the model to distinguish between pairs with complex structures. For example, in Figure 4.1, both **fell**$_{e54}$ and **wounded**$_{e55}$ events were previously classified as subevents of **fatality**$_{e53}$, but after including the fine-grained labels, the model learned that the subevent class between **fatality**$_{e53}$ and both **fell**$_{e54}$ and **wounded**$_{e55}$ is *NoRel* since the latter events are *BPast* events.

*In another **fatality**$_{e53}$, a Spanish military adviser, Gonzalo Perez Garcia, who **fell**$_{e54}$ into a coma after being seriously **wounded**$_{e55}$ in a shootout last month died Wednesday, the Spanish Defense Ministry said.*

Figure 4.1: Example of a text in which the relationship between one event and two other events is mis-classified without the fine-grained Foreground/Background feature.

4.3 Event Co-reference Task

Event co-reference is the task of determining whether two events refer to the same event in the real world. That is, given a pair of events, a system should determine whether the two events corefer. For this experiment, I trained a pairwise logistic regression model over the features shown in Table 4.2.

Feature	Representation	Description
Major POS	One-hot	The major POS of e_i and e_j.
Tense	One-hot	The tense of e_i and e_j.
Aspect	One-hot	The aspect of e_i and e_j.
Semantic Frame	One-hot	The semantic frame of e_i and e_j using SEMAFOR (Das et al., 2010).
Discourse Relation	One-hot	The discourse relation between elementary discourse units (EDUs), where e_i or e_j are mentioned in using the Feng-Hirst discourse parser (Feng and Hirst, 2014) as explained earlier in Section 3.6.
Semantic Similarity	Numeric	The semantic similarity between e_i and e_j using BERT model *bert-base-uncased* (Devlin et al., 2019) and following the same procedure described in Section 3.6.
Co-referring Event Arguments	One-hot	Whether the arguments of e_i and e_j corefer. The arguments are extracted and resolved using `AllenNLP` SRL (Lee et al., 2017) and co-reference resolution systems (Gardner et al., 2018).

Table 4.2: Features used in event co-reference experiment. e_i and e_j represent the target pair.

	hyper-parameters		
Task	multi_class	solver	C
Subevent	ovr	liblinear	0.01
Event coref.	multinomial	lbfgs	0.1
Temporal	ovr	liblinear	0.0001

Table 4.3: The hyper-parameters used in all experiments corresponding to the scikit-learn's implementation of logistic regression.

I trained a pairwise logistic regression classifier from `scikit-learn` over the features using parameters shown in Table 4.3. I used the IC corpus and conducted 5-fold cross-validation for the experiment. As shown in Table 4.4, the pairwise performance increases by 2% after including the Foreground/Background fine-grained features. The first column in the table shows the performance of Liu et al. (2014) pairwise model on 65 documents of the IC corpus, which I use as a baseline. A shallow error analysis after including Foreground/Background fine-grained features reveals that almost all corrected cases were false negative, and the events involved are foreground events. This observation indicates that detecting foreground events could be a useful intermediate step for event co-reference improvement.

Model	**Prec.**	**Rec.**	**F$_1$**
Liu et al. (2014)	0.48	0.59	0.53
My System	0.52	0.84	0.65
+Fine-grained Labels	0.55	0.85	**0.67**

Table 4.4: Event co-reference experiment result.

4.4 Temporal Relation Extraction Task

Extracting temporal information from text is a challenging but important task in NLP. In this experiment, I target the extraction of the temporal relation between events, which is one of the fundamental tasks in temporal processing as identified in the series TempEval (TE) workshops (Verhagen et al., 2007, 2010; UzZaman et al., 2013) and the work of Cassidy et al. (2014). Cassidy et al. (2014) reduced the temporal relations from fourteen fine-grained relations (described in Section 1.4.3) to five coarse-grained relations (i.e., *before*, *after*, *includes*, *is-included*, and *simultaneous*). The argument for reducing and making coarse-grained relations is that this fine-grained distinction may complicate an already difficult task, and there is no clear benefit of the fine-grained distinction yet. These five coarse-grained relations are described with examples below:

- Before
    - An event is before the other event in time

  - Ex: She **vomited** shortly before **surgery**.
  - vomited *BEFORE* surgery

- After
  - The inverse of *Before* relation

- Includes
  - An event temporally includes the other event
  - Ex: During the **surgery**, a butterfly in Kashmir **flapped** its wings three times.
  - surgery *INCLUDES* flapped

- Is-included
  - The inverse of *Includes* relation

- Simultaneous
  - The two events have the same temporal boundaries and extent
  - Ex: She **listened** to music during her whole **drive** home.
  - listened *SIMULTANEOUS* drive

These partial temporal orderings can be then used to construct a complete temporal graph. Notably, *Includes* and *Is-included* relations are more general cases of the subevent relation introduced in Section 2.1. That is, my model introduced in Section 2.1 classifies a pair of events as *subevent* if one event is *spatiotemporally* contained by the other. For example, in this sentence: *During the* **surgery**$_{e56}$*, most of the tumor was* **excised**$_{e57}$, the **excised**$_{e57}$ event is a subevent of the **surgery**$_{e56}$ event. However, in this sentence: *During the* **surgery**$_{e58}$*, a butterfly in Kashmir* **flapped**$_{e59}$ *its wings three times*, the **flapped**$_{e59}$ event is not a subevent of the **surgery**$_{e58}$ event.

For this experiment, I used the recently published dataset TDDiscourse (Naik et al., 2019), an augmented dataset of TimeBank-Dense (Cassidy et al., 2014) focused on discourse-level temporal ordering and using the same set of temporal relations as TimeBank-Dense (i.e., *before*, *after*, *includes*, *is-included*, and *simultaneous*). This dataset focused on global discourse-level temporal ordering, which is suitable to measure the effectiveness of foreground and background events knowledge in modeling the discourse-level temporal relation extraction. The annotation of the TDDiscourse corpus consists of two sets: Manual annotation (TDD-Man) and Automatic inference (TDD-Auto) as shown in Table 4.5. I experiment on both.

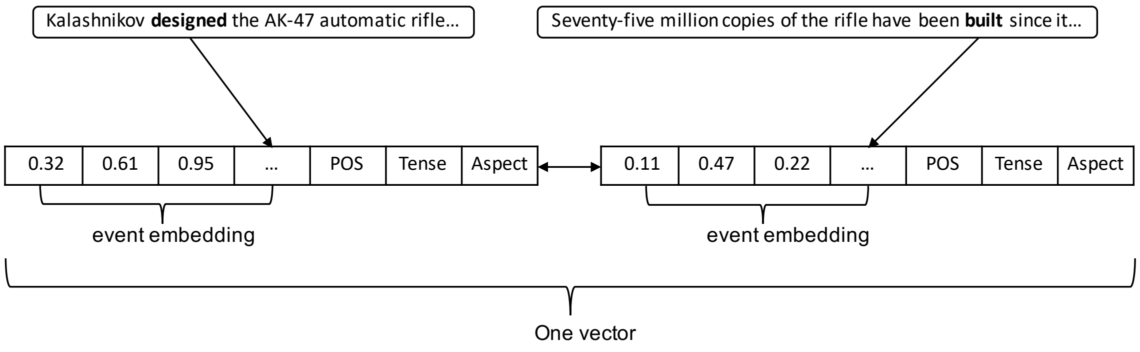

Kalashnikov **designed** the AK-47 automatic rifle…
Seventy-five million copies of the rifle have been **built** since it…
0.32
0.61
0.95
…
POS
Tense
Aspect
0.11
0.47
0.22
…
POS
Tense
Aspect
event embedding
event embedding
One vector

Figure 4.2: Concatenating pair's BERT embedding, POS, Tense, and Aspect as one vector.

Dataset	Train	Dev	Test
TDD-Man	4000	650	1500
TDD-Auto	32609	1435	4258

Table 4.5: TDDiscourse corpus Statistics. These numbers are copied from Naik et al.'s (2019) work.

I designed a simple and effective approach by concatenating pair's BERT embedding[1], POS, tense, and aspect as one vector, shown in Figure 4.2. I trained a logistic regression

[1]Event embedding is extracted the same way discussed in Section 3.6, Semantics.

classifier over these features using hyper-parameters shown in Table 4.3. I followed Naik et al. (2019) split setup of train, validation, and test sets and compared the performance of my model to all models reported on the corpus. Similar to my previous experiments, I add Foreground/Background fine-grained features to the model and measure the performance with and without these features.

	TDD-Auto			TDD-Man		
Model	Pre.	Rec.	F_1	Pre.	Rec.	F_1
MAJOR	34.2	32.3	33.2	37.8	36.3	37.1
CAEVO	61.1	32.6	42.5	32.3	10.7	16.1
BiLSTM	55.7	48.3	51.8	24.9	23.8	24.3
Ning et al. (2017)	46.4	45.9	46.1	23.9	23.8	23.8
My System	60.6	60.6	60.6	42.4	42.4	42.4
+Fine-grained Labels	61.2	61.2	**61.2**	42.9	42.9	**42.9**

Table 4.6: The first four models are an adaptation of state-of-the-art temporal models on TDD-Auto and TDD-Man reported by (Naik et al., 2019). The last two rows show my model without and with Foreground/Background fine-grained features, respectively.

As shown in Table 4.6, my approach, in general, outperforms all models on both TDD-auto and TDD-man by 9% and 5%, respectively. The reason behind the low performance of the other models has been addressed by Naik et al. (2019) and is out of scope for this thesis. With regard to my model, as shown in Table 4.6, adding Foreground/Background fine-grained features did not help much in improving the model performance. This is expected due to the fact that the fine-grained model was trained on a closed-domain (i.e., Intelligence Community (IC) news articles), which is a small fraction—55% are IC news articles, and 40% of these are broadcast news—in the TDDiscourse corpus test set.

CHAPTER 5

EVENT BASED FRAGMENTED STORY STITCHING

Understanding subevents and the distinction between foreground/background events are together potentially useful in new NLP tasks. In this chapter, I introduce the task of story fragment stitching, which is the process of automatically aligning and merging event sequences of partial tellings of a story (i.e., story fragments). This task is similar to the cross-document event co-reference relation task but more challenging because the overall timeline of the story's events need to be preserved across all fragments. For this problem, I introduce a graph-based unsupervised approach to align a set of story fragments into a full, ordered, end-to-end list of story events.

5.1 Introduction

Events are the building blocks for stories. Stories are found throughout our daily lives, e.g., in news, entertainment, education, religion, and many other domains. Understanding stories is a long-term goal of the field of artificial intelligence and NLP. (Charniak, 1972; Schank and Abelson, 1977; Wilensky, 1978; Dyer, 1983; Riloff, 1999; Frank et al., 2003; Mueller, 2007; Winston, 2014). Automatically understanding stories is beneficial in many NLP tasks, and more information can be extracted from stories, including concrete facts about events and people (Eisenberg and Finlayson, 2017).

One interesting and challenging task that relates to event relation extraction tasks and has not yet been solved is what I call *story fragment stitching*. In this task, I seek to merge partial tellings of a story—where each partial telling contains part of the sequence of events of a story, perhaps from different points of view, and may be found across different sources or media—into one coherent narrative, which may then be used as the basis for further processing. Conceptually, this task is similar to both cross-document event co-reference (CDEC) and event ordering in NLP. However, story fragment stitching, as I define it, presents a more challenging problem for at least two reasons. First, and unlike event co-reference, the overall timeline of the story's events needs to be preserved across all fragments. Second, and unlike event ordering, which targets only events related to a single entity, my work considers all events across all fragments.

I proposed an unsupervised approach to solving the problem of stitching a fragmented story. I apply this approach to a concrete example of this problem, namely, the story of the prophet Moses as found in the Quran, the Islamic holy book. The story of Moses is not found in one single telling in the Quran; rather, it is found in eight fragments spread across six different chapters (the chapters of the Quran are called *suras*), with the story comprising 7,931 total words across 283 verses that range from 2 to 94 words in length.

5.2 Task

I define the goal of story fragment stitching as aligning a set of story fragments into a full, ordered, end-to-end list of story events. I assume that the events in the story are presented in the chronological order in which the events of the story take place (i.e., the *fabula* time order) (Bordwell, 2007). That is, the story fragments are ordered lists of events, where the order is that of the fabula, namely the order of events as they happen in the story world. In many stories, the fabula order is different from the discourse order, but I do not consider this case here; I leave the problem of extracting the chronological order of events to other work.

I also assume that each fragment shares at least one event with another fragment. The output of the system is an ordered list of nodes, where each node is a collection of event mentions (corefering events) that all describe one particular event, and these nodes are in the same order as the overall fabula.

5.3 Related Work

The problems most closely related to story stitching are the problem of cross-document event co-reference (CDEC) and cross-document event ordering. In CDEC systems, the goal is to group expressions that refer to the same event across multiple documents (Bagga and Baldwin, 1999; Lee et al., 2012; Goyal et al., 2013; Saquete and Navarro-Colorado, 2017; Kenyon-Dean et al., 2018; Barhom et al., 2019).

Bagga and Baldwin (1999) proposed the first approach to this task; for each document, their system built a summary with respect to the entity of interest by selecting all sentences in which that entity appeared and computed that summary's similarity to summaries extracted from other documents. Pairs of summaries having similarity above a certain threshold were considered to be about the same event. Lee et al. (2012) sought to model entities and events jointly by using a linear regression model's decision to merge clusters of entities with each other and clusters of events with others, allowing information to be shared between the clusters through modeling semantic role dependencies.

In contrast, Goyal et al. (2013) used knowledge extracted via a syntax-based distributional semantic approach to detect event co-reference. They used extracted relations between words from Wikipedia articles. The assumption was that two mentions generally refer to the same event when their semantic information (i.e., actions, agents, patients, locations, and times) are (almost) the same. Therefore, when the semantic role labeling system fails to determine the event's roles, the system can fall back to the knowledge extracted from Wikipedia. Recently and similar to Lee et al.'s work, Barhom et al. (2019) proposed a neural network-based model that models entities and events jointly.

For the event ordering task, which was introduced in SemEval-2015 (Minard et al., 2015), the goal is to order cross-document events that involve a specific target entity.

That is, a system should produce a timeline for a specific target entity that consists of the ordered list of the events in which that entity participates.

Saquete and Navarro-Colorado (2017) proposed an approach to align the timelines of events between documents. They used the Temporal Information Processing System (TIPSem) to first extract temporal relationships between events at the document level and then cluster temporally compatible events between documents. They used two different distributional semantic models, LDA and Word2Vec embeddings, to measure the semantic compatibility between events.

Similar to the event ordering task, the within-document event sequence detection task, which was introduced in the TAC KBP 2017 event track (Mitamura et al., 2017), aims to identify the event sequence (i.e., after links) that occurs in a script (Schank and Abelson, 1977). Liu et al. (2018) proposed a graph-based approach to capture the order of events that occur in a script.

Despite this very interesting and useful prior work, all aforementioned systems are not directly applicable to the task of story fragment stitching as I define it. In particular, CDEC systems ignore the timeline of the story's events (i.e., the overall timeline of the story's events is not guaranteed to be preserved across all fragments), while event ordering systems only order certain events related to a specific target.

Researchers have explored several ways of assessing the similarity between stories (Schank and Abelson, 1975; Roth and Frank, 2012; Finlayson, 2012; Iyyer et al., 2016; Nikolentzos et al., 2017; Chaturvedi et al., 2018). These works provided valuable ways to capture the similarity between stories. However, the story similarity task is not directly applicable to the task of stitching fragmented stories, where the goal is to order events across multiple stories (fragments), except in the simple baseline multiple sequence alignment approach (Reiter, 2014). Multiple sequence alignment was built upon simpler two-sequence alignment approaches, of which the Needleman-Wunsch algorithm

(Needleman and Wunsch, 1970) is the prototypical approach, and involves a linear-time dynamic programming solution. The Needleman-Wunsch algorithm relies on a gap cost and a similarity function and works on two input sequences to produce a global alignment in which each element in both sequences is either linked or skipped. I considered the multiple sequence alignment approach (Reiter, 2014) as a baseline for this work.

5.4 Approach

I present an unsupervised approach to the story fragment stitching problem inspired by Finlayson (2016), which is in turn based on model merging, a regular grammar learning algorithm (Stolcke and Omohundro, 1993). My approach consists of two main components: model formulation (§5.4.1), and the graph merge to align fragments events into a full, ordered, end-to-end list of story events (§5.4.2).

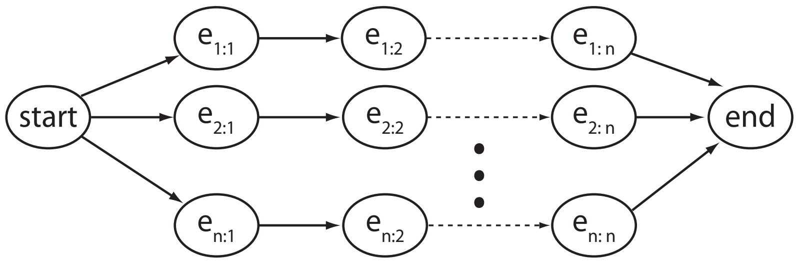


Figure 5.1: The initial model constructed using the Moses story fragments. Each node represents an event's vector. Each fragment generates its own linear branch running from the start node to the end node where *i* in $e_{i:j}$ represents the fragment's number, and *j* represents the event's number.

5.4.1 Model Formulation

The first step of my approach is model initialization, which is shown in Algorithm 2 lines 4–8. Using the function `constructLinearBranch`, I convert each fragment's list

Algorithm 2:

```
$F$ : set of text fragments $f$
$E$ : map of $f$ to sets $e_f$ of gold event annotations
// Create initial model
$G \leftarrow \emptyset$
foreach $f \in F$ do
    $g \leftarrow$ constructLinearBranch($f$, $E.get(f)$)
    $G.add(g)$
end
$model \leftarrow$ linkGraphs($G$)
// Merging process
$\alpha \leftarrow$ computeTFIDFAvgSim($F$)
$nodesSim \leftarrow$ computeNodesSim($model$)
$(maxPairSim,pairs) \leftarrow$ findMostSim($nodesSim$)
repeat
    if $\neg$ pairsIntroduceCycle($model, pairs$) then
        $model \leftarrow$ merge($pairs$)
        $nodesSim \leftarrow$ updateNodesSim($model,nodesSim$)
    else
        $nodesSim \leftarrow$ setSimToZero($pairs,nodesSim$)
    end
    $(maxPairSim,pairs) \leftarrow$ findMostSim($nodesSim$)
until $maxPairSim < \alpha$;
bestPath $\leftarrow$ findBestPath($model$)
```

of events into a linear directed graph (linear branch) in which each node contains only a single event.

Each event is represented by a vector that is a concatenation of the event contextualized embedding from the BERT model and *tf-idf* weights of the event lemma and its semantic arguments. The *tf-idf* is the standard term weighting approach to reflect how important a word is in a document in comparison to the rest of the documents (Salton and McGill, 1986). Using the function `linkGraphs`, I link all linear branches to a start and an end node, resulting in one directed graph of the whole set of fragments, as shown in Figure 5.1.

This initial model will be used to generate possible solutions by merging different nodes on the basis of a similarity measure, discussed below. When two nodes A and B are merged, the new node C should contain an average vector of both A and B. In the next section, I introduce the merge approach.

5.4.2 Graph Merge

The second step of my approach is model merging, shown in Algorithm 2 lines 10–25. I first compute a threshold α using the `computeTFIDFAvgSim` function, which takes the average of the highest and lowest cosine similarity values between all fragments using *tf-idf* weights. α sets the minimum similarity required to merge two nodes; for our data α was 0.39. Next, using a cosine similarity measure, the `computeNodesSim` function computes the full set of similarity scores between all pairs of nodes. Then the algorithm starts by searching for the most similar nodes using the `findMostSim` function, lines 14 and 23, and merges the most similar nodes using the `merge` function. Because the fragments are assumed to be already in fabula time order, the `pairsIntroduceCycle` boolean function disallows merges that would introduce cycles, ignoring (and removing) self-loops (the *no-cycles* constraint), and thereby preserves the overall order of the events. Note that disallowing cycles also prevents merges of non-neighboring nodes within the same fragment. The new merged node then contains a weighted average vector of the old nodes' vectors, and nodes' similarity are updated using the `updateNodesSim` function. The algorithm continues to merge nodes until the similarity measure drops below α. Because the final resulting graph is not guaranteed to contain only one path from start to end, by using the `bestPath` function, the path with the maximum number of merged nodes (based on the number of events) is considered to be the final output of the model.

Sura	Verses	# of Verses	# of Event Mentions	# of Moses Event Mentions	# of Event Categories
2. Al-Baqarah	50–60, 63–73, 92–93	13 (11+2)	36 (25+11)	9 (6+3)	6
7. Al-A'raf	103–161	59	149	99	23
10. Yunus	75–92	18	36	12	6
20. Ta-Ha	9–98	90	195	78	28
26. Ash-Shuara	10–67	58	63	37	8
28. Al-Qasas	7–40	34	94	66	14
Total	283	272	573	301	

Table 5.1: Number of verses (inclusive ranges), event mentions, and events of the Moses story in each fragment. Listed are the total number of non-Reporting Event mentions, the total number of event mentions labeled as an event from the Moses timeline, and the total number of distinct labels found in that fragment. The first fragment (Al-Baqarah verses 50–60) is omitted from the data because it violated the linear time order constraint.

5.5 Data

Moses was an important figure whose story is central to the major Abrahamic religions, including Judaism, Christianity, and Islam. Moses' story is found in fragmentary form throughout the holy books of these religions, with some parts repeated, but in different contexts and sometimes from different perspectives. In the Quran, the holy book of Islam, the story of Moses appears in eight different fragments across six different chapters (suras) comprising 283 verses. Thus the story of Moses serves as an excellent example for the evaluation of my approach to story fragment stitching. The relevant suras and verses are listed in Table 5.1, along with the number of events present in the fragments of each chapter.

The annotation of the data is part of Asgari's thesis (Asgari, 2014). Three annotators annotated verses based on a comparative analysis of Moses' story in the Old Testament and the Quran by (Ghanbari and Ghanbari, 2008). The Ghanbari study breaks Moses' story into 43 event categories, shown in Table 5.2 in chronological order. The annotators labeled each verse with its single relevant event and reported an agreement of 0.76 Fleiss' kappa, which represents excellent agreement. The annotation was originally done on the Arabic version of the Quran, but they transferred the annotations to an English translation (Ali, 1973) for the remainder of the study.

I excluded one fragment (Sura 2 [Al-Baqarah], verses 50–60) from the analysis because its timeline is quite different from the fabula order. I manually extracted 708 total event mentions from the remaining seven fragments. My annotation procedure followed the standards outlined for events in the TimeML standard (Saurı et al., 2006). I omitted 135 *Reporting* mentions (e.g., *say*, *reply*, etc.) because these usually are just indicators of direct speech and do not correspond to plot events. This resulted in 573 event mentions relevant to the plot, which I labeled as to which specific event it referred in the Moses timeline (Table 5.2). Of these, 301 mentions were labeled with an event described in the timeline, while 272 were not relevant.

Event	**# frags.**	**# event**	**F1**
Moses's Birth			
1. Moses' Birth and left in the Nile.	2	6	0.33
Moses is Rescued from the Nile			
2. Moses is rescued from the Nile.	2	4	0.45
3. Moses' sister kept an eye on him.	2	4	0.55
4. Moses brought back to his mother.	2	2	0.34
5. Moses after infancy and through maturity.	1	2	0.68
Moses kills the Egyptian			
6. Moses beats and kills the Egyptian.	2	10	0.85
Moses flees to the Madyan			
7. Moses ran away to the Madyan.	1	1	0.74

Continued on next page

Table 5.2 – continued from previous page

Event	# frags.	# event	F1
Moses' Marriage			
8. Moses protected Shu'ayb's daughters.	1	20	0.53
9. Moses traveled with his family.	2	3	0.50
Moses is Chosen to be a Prophet			
10. Moses saw the fire from the distance.	2	12	0.60
11. Moses talked to God through the burning bush.	2	4	0.89
God Shows Moses the Miracles			
12. God changed the wand to the snake.	2	11	0.69
13. God illuminated Moses' hand.	2	5	0.63
God Sends Moses to the Pharaoh			
14. God commanded Moses to meet the Pharaoh.	2	9	0.55
Moses Speaks with the Pharaoh			
15. Moses and Aaron went to the Pharaoh with miracles.	4	8	0.47
16. Moses showed the Pharaoh the signs.	3	10	0.56
17. The Pharaoh refused their message.	3	4	0.55
18. The Pharaoh accused Moses.	4	5	0.80
19. The Pharaoh requested a competition with Moses.	3	8	0.68
20. Competition between Moses and the magicians.	4	34	0.63
21. Magicians believed in Moses's message.	3	8	0.88
22. Magicians are threatened by the Pharaoh.	3	6	0.72
23. The Pharaoh's cruelty to the believers.	1	6	0.28
God Sends Calamities to Egypt			
24. Calamities are sent to the Egyptians and the Pharaoh.	1	6	0.58

Continued on next page

Table 5.2 – continued from previous page

Event	# frags.	# event	F1
25. God withdraws the punishment.	1	1	0.65
26. God commanded Moses to travel with his people.	3	6	0.31
27. The Pharaoh and his army followed Moses and his people.	3	6	0.63
Parting of the Red Sea			
28. Separation of the Sea and drowning of the Pharaoh.	5	14	0.49
29. God saved Moses and his people.	3	5	0
Going to Mt. Sinai to Receive the Commandments			
30. Moses went to Sinai for 40 nights.	3	6	0.47
31. God sent down food and brings forth water.	1	4	0.73
32. Moses met God who appeared on the mountain.	2	15	0.43
33. Moses delivered the commands and the stone tablets.	3	3	0
The People Betray God			
34. Worshipping the Calf in the Absence of Moses.	2	14	0.19
35. Moses returned to his people.	1	3	0.29
36. Samiri explained to Moses what he saw.	2	3	0
37. Moses blamed his brother.	1	10	0
38. Moses returned to God.	1	3	0
39. Moses stroked the stone.	1	9	0
Wandering in the Desert			
40. Israelites are commanded to take over the holy region.	1	5	0
41. The disobedient Israelites won't enter the holy region.	2	4	0
42. God punished them.	2	1	0

Continued on next page

Table 5.2 – continued from previous page

Event	# frags.	# event	F1
43. Sacrifice of a heifer.	1	1	0

Table 5.2: The 43 events in the Moses timeline. The second column refers to the number of fragments in which the corresponding event appears. The third column refers to the number of mentions of the event across all fragments. The last column is the standard F_1 measure for the extraction of the corresponding event, compared to the gold standard.

5.6 Experiment

I evaluate my approach against a gold-standard annotation of Moses' story from the Quran. I first demonstrate the experiment setup (§5.6.1) and the evaluation (§5.6.2). Then I report the performance of my approach (§5.6.4). Finally, I discuss the error analysis of the performance of my system (§5.7).

5.6.1 Experimental Setup

I used the `netwrokx` library (Hagberg et al., 2008) for graph operations. I extracted event contextualized embedding using the `flair` implementation (Akbik et al., 2018) of the BERT model with the default parameters[1]. The *tf-idf* weights for the lemmas of all tokens excluding stop words are computed using `spaCy` (Honnibal and Montani, 2017) and `scikit-learn` libraries (Pedregosa et al., 2011). The event arguments are extracted and resolved using `AllenNLP` SRL and co-reference systems (Gardner et al., 2018; He et al., 2017; Lee et al., 2017).

[1]`bert_base_uncased, layers=-1, pooling_operation=first`

5.6.2 Evaluation

For the evaluation, I used the temporal awareness measure (UzZaman et al., 2013) used in both event ordering task SemEval-2015 (Minard et al., 2015) and event sequence task TAC-KBP-2017 (Mitamura et al., 2017). The temporal awareness metric calculates precision and recall values based on the closure and reduction graphs. For a directed graph, a reduced graph is derived from the original graph by having the fewest possible edges that have the same reachability relation as the original graph (Liu et al., 2018). In this work, the final directed path of nodes in the final model represents the reduced graph. For example, consider the final directed path of nodes in the final model to be the following:

$$start \rightarrow n_1 \rightarrow n_2 \rightarrow n_3 \rightarrow end$$

where, for example, events $e_1, e_2 \in n_1$, $e_3 \in n_2$, and $e_4, e_5 \in n_3$. The reduced graph (G^-) is represented as the following edges: $\langle(e_1, e_3), (e_2, e_3), (e_3, e_4), (e_3, e_5)\rangle$ and the transitive closure graph (G^+) is represented as the following edges: $\langle(e_1, e_3), (e_2, e_3), (e_1, e_4), (e_1, e_5),$ $(e_2, e_4), (e_2, e_5), (e_3, e_4), (e_3, e_5)\rangle$, where the relation between (e_i, e_j) is defined as the *before* relation. The temporal awareness metric calculates the precision and recall as follows:

$$precision = \frac{|System^- \cap Reference^+|}{|System^-|} \tag{5.1}$$

$$recall = \frac{|Reference^- \cap System^+|}{|Reference^-|} \tag{5.2}$$

,where *System* and *Reference* are the proposed approach and the gold standard, respectively. The final F_1 score is the harmonic mean of the precision and recall values.

Model	Prec.	Recall	F_1
Needleman-Wunsch	0.41	**0.70**	0.52
tf-idf	0.43	0.40	0.42
BERT	0.77	0.50	0.61
Concat	**0.81**	0.51	**0.63**

Table 5.3: Results on the Moses data using the F_1 temporal awareness metric. Concat is the proposed model as described in 5.4.1, whereas tf-idf and BERT are variant models of the proposed model when tested alone.

5.6.3 Baseline

I used the Needleman-Wunsch algorithm (Needleman and Wunsch, 1970), a well-known global alignment algorithm used in bioinformatics, as a baseline. Using dynamic programming, this algorithm searches for the optimal alignment of multiple sequences (the events lemma in my case) by using a scoring function that penalizes the dissimilarities and the insertion of gaps. I used the default implementation[2] developed by Dekker and Middell (2011), which follows the group of progressive alignment algorithms where two sequences are aligned, and then the result is aligned to the next sequence. It repeats the procedure until all sequences are aligned.

5.6.4 Result

Table 5.3 shows my model results compared to the baseline. In the table, I compare three models: the proposed model **Concat**, *tf-idf*, and BERT, the latter two are sub-models of the proposed model when considered alone for graph's nodes vector representation as described in Section 5.4.1. As shown in bold in Table 5.3, the **Concat** approach achieves

[2]`https://github.com/interedition/collatex`

0.63 F_1 which outperforms the baseline by 11 points and both *tf-idf* and BERT alone by 21 and 2 points, respectively. Also, the table shows that concatenating both *tf-idf* and BERT produced the best result even though *tf-idf* alone underperformed the baseline. It is also clear that BERT contextualized embeddings play a major role in the model merging approach for nodes' vector representation when assessing similarity between nodes.

5.7 Error Analysis and Discussion

Inspection of the results revealed several sources of errors, aside from the usual noise introduced by the various sub-components, such as the SRL or co-reference systems. Some peculiarities of Quranic language cause errors. For example, the word *We* is usually present as an event's argument when God is speaking of himself. This causes problems for the co-reference resolution system in that it does not pair *we* with mentions such as *Lord* and *God*, and thus introduces additional errors into the system. Some events also have the same event mention and arguments but happened at different points in the timeline. Example 5.2 shows text from different parts of the story: the first is *when God shows Moses one of the signs*, whereas the second is *when Moses shows the Pharaoh the sign*. Notably, the two events have the same event triggers (showed in bold) and the same arguments (underlined).

(20:19–20) "Throw it down, O Moses," said (the Voice). So he ***threw*** *it down, and lo, it* ***became*** *a running serpent.*

(7:106–107) He said: "If you have brought a sign then display it, if what you say is true." At this Moses ***threw*** *down his staff, and lo, it* ***became*** *a live serpent.*

Figure 5.2: An example of two events at different points in the timeline.

Further, my approach is sensitive to the order of merges. If an incorrect merge is performed early, this can eliminate correct merges later on account of the *no-cycles* constraint. Therefore performing only the highest confidence merges first is critical, and errors in that process degrade other distance parts of the model.

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this thesis, I addressed four problems with state-of-the-art approaches to understanding events and their relations (§2, §3, and §4), and proposed a novel solution to the problem of stitching story fragments (§5). Below I list the conclusion of this thesis.

- **Subevent Detection** (Chapter §2). I presented a model to detect subevent relations in news articles that outperforms two prior approaches to this task. My model involves several novel discourse and narrative features, as well as a number of feature modifications. Also, I performed an extensive error analysis, e.g., I showed that having two event hierarchies in the same sentence is a major problem, as well as having a significant separation between parent and child events.

- **Foreground and Background Event Detection** (Chapter §3). I presented a novel task: distinguishing between foreground and background events, as well as marking the general temporal position of background events relative to the foreground period. I provided an annotated dataset and built a featurised logistic regression model that performs well on this task and that relies heavily on discourse understanding. I showed that while my model's performance is reasonable, there is still room for improvement by the introduction of commonsense or world knowledge to aid in reasoning.

- **Integrating Foreground and Background Events into Event Relation Detection** (Chapter §4). I showed the effectiveness of using foreground and background events knowledge in modeling event relations, namely, subevent detection, event co-reference, and discourse-level temporal relation extraction. That is, for each task, I built a system with including the foreground and background events knowledge, produced by my system in Chapter 3, as a feature. The result of this integration shows that detecting foreground and background events is very useful in modeling event relations.

- **Event-Based Fragmented Story Stitching** (Chapter §5). I introduced the story fragment stitching problem: the task of merging partial tellings of a story into a unified whole. I introduced a novel approach that models the story's fragments in a graph and applies an adapted model merging approach to merge similar nodes and produce an ordered, end-to-end list of story events.

For future work, there is still room for improvement of all of my previous models. More precisely, in modeling subevent relations, one can incorporate domain knowledge or lexical resources such as an ontology that captures certain relations between events. For example, the Rich Event Ontology (Brown et al., 2017) is one of the ontologies that identified the *hasSubevent* relation between some events such as the **verdict** event that always happens within a **trial** event. For the foreground and background events model, one can improve it by incorporating commonsense knowledge or external databases that contain information about certain events such as the Gulf War. This can be useful in various ways, e.g., retrieving some information about an event such as where or when the event happened because writers sometimes assume these are known by readers. Moreover, the foreground and background model can also be used to improve other event relations such as causal relations. Furthermore, I have developed a baseline model for temporally aligning cross-fragment events that can be used in a more generalized setting. That is, the stitching fragment model can be adopted and generalized on fragmented news articles such as the news articles about the Egyptian revolution of 2011; both subevent and foreground and background events models can be useful in this regard. This will be included in future work.

BIBLIOGRAPHY

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

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649.

Abdullah Yusuf Ali. 1973. *The Holy Qur'an: text, translation and commentary*. Islamic University of Al ima Mohammad ibn SAUD.

James Allan, Jaime G Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. 1998. Topic detection and tracking pilot study final report.

James F Allen. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

Jun Araki, Zhengzhong Liu, Eduard H Hovy, and Teruko Mitamura. 2014. Detecting subevent structure for event coreference resolution. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC)*, pages 4553–4558, Lisbon, Portugal.

Ehsaneddin Asgari. 2014. Topic model story merging algorithm. Master's thesis, Swiss Federal Institute of Technology - Lausanne (EPFL).

Emmon Bach. 1986. The algebra of events. *Linguistics and philosophy*, 9(1):5–16.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *The first international conference on language resources and evaluation workshop on linguistics coreference*, volume 1, pages 563–566. Citeseer.

Amit Bagga and Breck Baldwin. 1999. Cross-document event coreference: Annotations, experiments, and observations. In *Proceedings of the Workshop on Coreference and its Applications*, pages 1–8. Association for Computational Linguistics.

Shany Barhom, Vered Shwartz, Alon Eirew, Michael Bugert, Nils Reimers, and Ido Dagan. 2019. Revisiting joint modeling of cross-document entity and event coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4179–4189, Florence, Italy. Association for Computational Linguistics.

Cosmin Bejan and Sanda Harabagiu. 2010. Unsupervised event coreference resolution with rich linguistic features. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1412–1422, Uppsala, Sweden. Association for Computational Linguistics.

Parminder Bhatia, Yangfeng Ji, and Jacob Eisenstein. 2015. Better document-level sentiment analysis from RST discourse parsing. In *Proceedings of the 20th Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 2212–2218, Lisbon, Portugal.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Claire Bonial, Olga Babko-Malaya, Jinho D Choi, Jena Hwang, and Martha Palmer. 2010. Propbank annotation guidelines. *Center for Computational Language and Education Research, CU-Boulder*.

David Bordwell. 2007. *Poetics of Cinema*. New York: Routledge.

T Briscoe, A Copestake, and B Boguraev. 1990. Lexical semantics via lexicology. In *Proceedings of the 13th International Conference on Computational Linguistics, Helsinki*.

Susan Brown, Claire Bonial, Leo Obrst, and Martha Palmer. 2017. The rich event ontology. In *Proceedings of the Events and Stories in the News Workshop*, pages 87–97, Vancouver, Canada. Association for Computational Linguistics.

Lauri Carlson. 1981. Aspect and quantification, syntax and semantics.

Lynn Carlson and Daniel Marcu. 2001. Discourse tagging reference manual. *ISI Technical Report ISI-TR-545*.

Taylor Cassidy, Bill McDowell, Nathanel Chambers, and Steven Bethard. 2014. An annotation framework for dense event ordering. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA.

Eugene Charniak. 1972. *Toward a model of children's story comprehension.* Ph.D. thesis, Massachusetts Institute of Technology.

Snigdha Chaturvedi, Shashank Srivastava, and Dan Roth. 2018. Where have i heard this story before? identifying narrative similarity in movie remakes. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 673–678.

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

Zheng Chen and Heng Ji. 2009. 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. Association for Computational Linguistics.

Timothy Chklovski and Patrick Pantel. 2004. VerbOcean: Mining the web for fine-grained semantic verb relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 33–40, Barcelona, Spain.

Prafulla Kumar Choubey, Kaushik Raju, and Ruihong Huang. 2018. Identifying the most dominant event in a news article by mining event coreference relations. In *Proceedings of the 16th Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2018)*, pages 340–345, New Orleans, LA.

Dan Cristea, Nancy Ide, and Laurent Romary. 1998. Veins theory: A model of global discourse cohesion and coherence. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (ACL-ICCL)*, pages 281–285, Montreal, Canada.

Agata Cybulska and Piek Vossen. 2014. 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. European Language Resources Association (ELRA).

Dipanjan Das, Nathan Schneider, Desai Chen, and Noah A. Smith. 2010. Probabilistic frame-semantic parsing. In *Proceedings of Human Language Technologies: The 8th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2010)*, pages 948–956, Los Angeles, CA.

Ronald H Dekker and Gregor Middell. 2011. Computer-supported collation with collatex: managing textual variance in an environment with varying requirements. *Supporting Digital Humanities*, pages 17–18.

Leon Derczynski and Robert Gaizauskas. 2010. USFD2: Annotating temporal expresions and tlinks for tempeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation (SemEval'10)*, pages 337–340, Los Angeles, CA.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 14th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, pages 4171–4186, Minneapolis, MN.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, volume 2, pages 837–840. Lisbon.

Greg Durrett and Dan Klein. 2014. A joint model for entity analysis: Coreference, typing, and linking. *Transactions of the Association for Computational Linguistics*, 2:477–490.

Michael George Dyer. 1983. *In-depth understanding: A computer model of integrated processing for narrative comprehension*. MIT press.

Joshua Eisenberg and Mark Finlayson. 2017. A simpler and more generalizable story detector using verb and character features. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2708–2715.

Vanessa Wei Feng and Graeme Hirst. 2014. A linear-time bottom-up discourse parser with constraints and post-editing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 511–521, Baltimore, Maryland. Association for Computational Linguistics.

Xiaocheng Feng, Lifu Huang, Duyu Tang, Heng Ji, Bing Qin, and Ting Liu. 2016. A language-independent neural network for event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 66–71, Berlin, Germany. Association for Computational Linguistics.

Charles J Fillmore, Christopher R Johnson, and Miriam RL Petruck. 2003. Background to framenet. *International journal of lexicography*, 16(3):235–250.

Mark Alan Finlayson. 2016. Inferring propp's functions from semantically annotated text. *The Journal of American Folklore*, 129(511):55–77.

Mark Mark Alan Finlayson. 2012. *Learning narrative structure from annotated folktales*. Ph.D. thesis, Massachusetts Institute of Technology.

Edward M. Forster. 1927. *Aspects of the Novel*. E. Arnold & Co., London.

Stefan L Frank, Mathieu Koppen, Leo GM Noordman, and Wietske Vonk. 2003. Modeling knowledge-based inferences in story comprehension. *Cognitive Science*, 27(6):875–910.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia.

Bakhshali Ghanbari and Zohreh Ghanbari. 2008. Comparative study of moses' position in quran and torah. *Journal of Theology*, (5):73–90.

Goran Glavaš and Jan Šnajder. 2014. Constructing coherent event hierarchies from news stories. In *Proceedings of the Workshop on Graph-based Methods for Natural Language Processing (TextGraphs-9)*, pages 34–38, Doha, Qatar.

Goran Glavaš, Jan Šnajder, Parisa Kordjamshidi, and Marie-Francine Moens. 2014. Hieve: A corpus for extracting event hierarchies from news stories. In *Proceedings of 9th Language Resources and Evaluation Conference (LREC)*, pages 3678–3683.

Kartik Goyal, Sujay Kumar Jauhar, Huiying Li, Mrinmaya Sachan, Shashank Srivastava, and Eduard Hovy. 2013. A structured distributional semantic model for event coreference. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 467–473.

Joseph E Grimes, Roy E Grimes, and Joseph Evans Grimes. 1975. *The Thread of Discourse*. Walter de Gruyter, New York.

Ralph Grishman, David Westbrook, and Adam Meyers. 2005. Nyu's english ace 2005 system description. *ACE*, 5.

Aric Hagberg, Pieter Swart, and Daniel S Chult. 2008. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

Hilda Hardy, Vika Kanchakouskaya, and Tomek Strzalkowski. 2006. Automatic event classification using surface text features. In *Proc. AAAI06 Workshop on Event Extraction and Synthesis*, pages 36–41.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 473–483, Vancouver, Canada.

Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1127–1136, Portland, Oregon, USA. Association for Computational Linguistics.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. `https://github.com/explosion/spaCy`; Last accessed on Jun , 2020.

Eduard Hovy, Teruko Mitamura, Felisa Verdejo, Jun Araki, and Andrew Philpot. 2013. Events are not simple: Identity, non-identity, and quasi-identity. In *Proceedings of the Workshop on Events: Definition, Detection, Coreference, and Representation*, pages 21–28, Atlanta, GA.

Ruihong Huang, Ignacio Cases, Dan Jurafsky, Cleo Condoravdi, and Ellen Riloff. 2016. Distinguishing past, on-going, and future events: The eventstatus corpus. In *Proceedings of the 21st Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 44–54, Austin, TX.

Silja Huttunen, Roman Yangarber, and Ralph Grishman. 2002. Complexity of event structure in ie scenarios. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*, pages 1–7, Taipei, Taiwan.

Mohit Iyyer, Anupam Guha, Snigdha Chaturvedi, Jordan Boyd-Graber, and Hal Daumé III. 2016. Feuding families and former friends: Unsupervised learning for dynamic fictional relationships. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1534–1544.

Pustejovsky James. 1995. The generative lexicon.

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

Heng Ji, Joel Nothman, Ben Hachey, et al. 2014. Overview of tac-kbp2014 entity discovery and linking tasks. In *Proc. Text Analysis Conference (TAC2014)*, pages 1333–1339.

Shafiq Joty, Giuseppe Carenini, and Raymond T Ng. 2015. CODRA: A novel discriminative framework for rhetorical analysis. *Computational Linguistics*, 41(3):385–435.

Shunsuke Kamijo, Yasuyuki Matsushita, Katsushi Ikeuchi, and Masao Sakauchi. 2000. Traffic monitoring and accident detection at intersections. *IEEE transactions on Intelligent transportation systems*, 1(2):108–118.

Kian Kenyon-Dean, Jackie Chi Kit Cheung, and Doina Precup. 2018. Resolving event coreference with supervised representation learning and clustering-oriented regularization. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 1–10, New Orleans, Louisiana. Association for Computational Linguistics.

Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. 2008. A large-scale classification of english verbs. *Language Resources and Evaluation*, 42(1):21–40.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

Angeliki Lazaridou, Ivan Titov, and Caroline Sporleder. 2013. A Bayesian model for joint unsupervised induction of sentiment, aspect and discourse representations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL), Volume I*, pages 1630–1639, Sofia, Bulgaria.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. 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. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Beth Levin. 1993. English verb classes and alternation. *A preliminary investigation*.

F Li, H Sheng, and D Zhang. 2002. Event pattern discovery from the stock market bulletin', discovery science: 5th int. Conference.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 789–797, Uppsala, Sweden. Association for Computational Linguistics.

Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning (ICML)*, pages 296–304, San Francisco, CA.

Shulin Liu, Yubo Chen, Shizhu He, Kang Liu, and Jun Zhao. 2016. Leveraging framenet to improve automatic event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), Volume I*, pages 2134–2143, Berlin, Germany.

Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1789–1798, Vancouver, Canada. Association for Computational Linguistics.

Zhengzhong Liu, Jun Araki, Eduard Hovy, and Teruko Mitamura. 2014. 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. European Language Resources Association (ELRA).

Zhengzhong Liu, Teruko Mitamura, and Eduard Hovy. 2018. Graph based decoding for event sequencing and coreference resolution. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3645–3657, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jing Lu and Vincent Ng. 2017. Joint learning for event coreference resolution. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 90–101, Vancouver, Canada. Association for Computational Linguistics.

Jing Lu and Vincent Ng. 2018. Event coreference resolution: A survey of two decades of research. In *IJCAI*, pages 5479–5486.

Jing Lu, Deepak Venugopal, Vibhav Gogate, and Vincent Ng. 2016. 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. The COLING 2016 Organizing Committee.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Xiaoqiang Luo, Sameer Pradhan, Marta Recasens, and Eduard Hovy. 2014. An extension of BLANC to system mentions. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 24–29, Baltimore, Maryland. Association for Computational Linguistics.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Mstislav Maslennikov and Tat-Seng Chua. 2007. A multi-resolution framework for information extraction from free text. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL 2007)*, pages 592–599, Prague, Czech Republic.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, and Armand Puhrsch, Christian and-Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the 11th Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Anne-Lyse Minard, Manuela Speranza, Eneko Agirre, Itziar Aldabe, Marieke van Erp, Bernardo Magnini, German Rigau, and Rubén Urizar. 2015. SemEval-2015 task 4: TimeLine: Cross-document event ordering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 778–786, Denver, Colorado. Association for Computational Linguistics.

Paramita Mirza and Sara Tonelli. 2016. CATENA: CAusal and TEmporal relation extraction from NAtural language texts. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 64–75, Osaka, Japan. The COLING 2016 Organizing Committee.

Teruko Mitamura, Zhengzhong Liu, and Eduard H Hovy. 2015. Overview of TAC KBP 2015 event nugget track. In *Proceedings of the 8th Text Analysis Conference (TAC 2015)*, Gaithersburg, MD.

Teruko Mitamura, Zhengzhong Liu, and Eduard H. Hovy. 2017. Events detection, coreference and sequencing: What's next? overview of the tac kbp 2017 event track. In *TAC*.

Alexander PD Mourelatos. 1978. Events, processes, and states. *Linguistics and philosophy*, 2(3):415–434.

Erik T Mueller. 2007. Understanding goal-based stories through model finding and planning. In *Intelligent Narrative Technologies: Papers from the AAAI Fall Symposium*, pages 95–101. AAAI Press Menlo Park, CA.

Aakanksha Naik, Luke Breitfeller, and Carolyn Rose. 2019. Tddiscourse: A dataset for discourse-level temporal ordering of events. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 239–249.

Saul B Needleman and Christian D Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453.

Arne Neumann. 2015. discoursegraphs: A graph-based merging tool and converter for multilayer annotated corpora. In *Proceedings of the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015)*, pages 309–312, Vilnius, Lithuania.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. 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. Association for Computational Linguistics.

Giannis Nikolentzos, Polykarpos Meladianos, François Rousseau, Yannis Stavrakas, and Michalis Vazirgiannis. 2017. Shortest-path graph kernels for document similarity. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1890–1900.

Qiang Ning, Zhili Feng, and Dan Roth. 2017. A structured learning approach to temporal relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1027–1037, Copenhagen, Denmark. Association for Computational Linguistics.

Tim O'Gorman, Kristin Wright-Bettner, and Martha Palmer. 2016. Richer event description: Integrating event coreference with temporal, causal and bridging annotation. In *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*, pages 47–56, Austin, TX.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 392–402, Austin, Texas. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. 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. Association for Computational Linguistics.

Sameer Pradhan, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng, and Michael Strube. 2014. Scoring coreference partitions of predicted mentions: A reference implementation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 30–35, Baltimore, Maryland. Association for Computational Linguistics.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. 2008. The penn discourse treebank 2.0. In *LREC*. Citeseer.

Rashmi Prasad, Aravind Joshi, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, and Bonnie Webber. 2005. The Penn Discourse TreeBank as a resource for natural language generation. In *Proceedings of the Corpus Linguistics Workshop on Using Corpora for Natural Language Generation*, pages 25–32, Birmingham, UK.

James Pustejovsky, José Castaño, Robert Ingria, Roser Saurí, Robert Gaizauskas, Andrea Setzer, and Graham Katz. 2003a. TimeML: robust specification of event and temporal expressions in text. In *Fifth International Workshop on Computational Semantics (IWCS-5)*, pages 1–11.

James Pustejovsky, José M Castano, Robert Ingria, Roser Sauri, Robert J Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R Radev. 2003b. Timeml: Robust specification of event and temporal expressions in text. *New directions in question answering*, 3:28–34.

James Pustejovsky, Jessica Littman, Roser Saurí, and Marc Verhagen. 2006. Timebank 1.2 documentation. *Event London, no. April*, pages 6–11.

Marta Recasens and Eduard Hovy. 2011. Blanc: Implementing the rand index for coreference evaluation. *Natural Language Engineering*, 17(4):485–510.

Nils Reiter. 2014. *Discovering Structural Similarities in Narrative Texts using Event Alignment Algorithms*. Ph.D. thesis, Heidelberg University.

Ellen Riloff. 1999. Information extraction as a stepping stone toward story understanding. *Understanding language understanding: Computational models of reading*, pages 435–460.

William David Ross et al. 1924. *Aristotle's Metaphysics: Books ZN*, volume 2. Clarendon Press.

Michael Roth and Anette Frank. 2012. Aligning predicates across monolingual comparable texts using graph-based clustering. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 171–182. Association for Computational Linguistics.

Gilbert Ryle. 1949. The concept of mind: Hutchinson's university library, london, 1949. *RyleThe Concept of Mind1949*.

Gerard Salton and Michael J McGill. 1986. *Introduction to modern information retrieval*. McGraw-Hill, Inc., New York City.

Evan Sandhaus. 2008. *The New York Times Annotated corpus*. Linguistic Data Consortium, Philadelphia, PA. LDC Catalog Number LDC2008T19.

Estela Saquete and Borja Navarro-Colorado. 2017. Cross-document event ordering through temporal relation inference and distributional semantic models. *Procesamiento del Lenguaje Natural*, 58:61–68.

Roser Sauri, Jessica Littman, Robert Gaizauskas, Andrea Setzer, and James Pustejovsky. 2006. TimeML annotation guidelines, version 1.2.1. `https://catalog.ldc.upenn.edu/docs/LDC2006T08/timeml_annguide_1.2.1.pdf`.

Roser Saurı, Jessica Littman, Bob Knippen, Robert Gaizauskas, Andrea Setzer, and James Pustejovsky. 2006. Timeml annotation guidelines version 1.2. 1.

Roger C Schank and Robert P Abelson. 1975. Scripts, plans, and knowledge. In *IJCAI*, volume 75, pages 151–157.

Roger C Schank and Robert P Abelson. 1977. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Hillsdale, NJ: Lawrence Erlbaum.

Swapna Somasundaran. 2010. *Discourse-level relations for Opinion Analysis*. Ph.D. thesis, University of Pittsburgh.

Andreas Stolcke and Stephen Omohundro. 1993. Hidden markov model induction by bayesian model merging. In *Advances in neural information processing systems*, pages 11–18.

Jannik Strötgen and Michael Gertz. 2013. Multilingual and cross-domain temporal tagging. *Language Resources and Evaluation*, 47(2):269–298.

Meihan Tong, Bin Xu, Shuai Wang, Yixin Cao, Lei Hou, Juanzi Li, and Jun Xie. 2020. Improving event detection via open-domain trigger knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5887–5897, Online. Association for Computational Linguistics.

Shyam Upadhyay, Christos Christodoulopoulos, and Dan Roth. 2016. "Making the News": Identifying noteworthy events in news articles. In *Proceedings of the 4th Workshop on Events*, pages 1–7, San Deigo, CA.

Naushad UzZaman and James Allen. 2011. Temporal evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 351–356, Portland, OR.

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9.

Zeno Vendler. 1957. Verbs and times. *The philosophical review*, 66(2):143–160.

Suzan Verberne, LWJ Boves, NHJ Oostdijk, and PAJM Coppen. 2007. Evaluating discourse-based answer extraction for why-question answering. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 735–736, Amsterdam, The Netherlands.

Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the fourth international workshop on semantic evaluations (SemEval-2007)*, pages 75–80.

Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. 2010. Semeval-2010 task 13: Tempeval-2. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 57–62.

GNA Vesey. 1964. Action, emotion and will. by kenny anthony. routledge and kegan paul, 1963. pp. 245. 25s. *Philosophy*, 39(149):277–278.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. *Linguistic Data Consortium, Philadelphia*, 57:45.

Chih-Ping Wei and Yen-Hsien Lee. 2004. Event detection from online news documents for supporting environmental scanning. *Decision Support Systems*, 36(4):385–401.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, et al. 2011. Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.

Robert Wilensky. 1978. Understanding goal-based stories. Technical report, YALE UNIV NEW HAVEN CONN DEPT OF COMPUTER SCIENCE.

Patrick Henry Winston. 2014. The genesis story understanding and story telling system a 21st century step toward artificial intelligence. Technical report, Center for Brains, Minds and Machines (CBMM).

Zhibiao Wu and Martha Palmer. 1994. Verbs semantics and lexical selection. In *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics (ACL)*, pages 133–138, Las Cruces, NM.

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 289–299, San Diego, California. Association for Computational Linguistics.

Hang Yang, Yubo Chen, Kang Liu, Yang Xiao, and Jun Zhao. 2018. DCFEE: A document-level Chinese financial event extraction system based on automatically labeled training data. In *Proceedings of ACL 2018, System Demonstrations*, pages 50–55, Melbourne, Australia. Association for Computational Linguistics.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5284–5294, Florence, Italy. Association for Computational Linguistics.

Xingxing Zhang, Mirella Lapata, Furu Wei, and Ming Zhou. 2018. Neural latent extractive document summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 779–784, Brussels, Belgium. Association for Computational Linguistics.

Deyu Zhou, Linsen Guo, and Yulan He. 2018. Neural storyline extraction model for storyline generation from news articles. 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 1727–1736, New Orleans, Louisiana. Association for Computational Linguistics.

VITA

MOHAMMED ALDAWSARI

December 13, 1987	Born, Wadi Ad-Dawasir, Riyadh
2011	B.S., Computer Science Prince Sattam Bin Abdulaziz University Al Kharj, Riyadh
2016	M.S., Computer Science DePaul University Chicago, Illinois

PUBLICATIONS AND PRESENTATIONS

Mohammed Aldawsari and Mark Finlayson. Detecting subevents using discourse and narrative features. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019), pages 4780–4790, Florence, Italy, 2019. URL `https://www.aclweb.org/anthology/P19-1471/`.

Deya Banisakher, W. Victor Yarlott, Mohammed Aldawsari, Naphtali Rishe, and Mark Finlayson. Improving the identification of the discourse function of news article paragraphs. In Proceedings of the First Joint Workshop on Narrative Understanding, Storylines, and Events, pages 17–25, Online, July 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.nuse-1.3/`

Mohammed Aldawsari, Ehsaneddin Asgari, and Mark Finlayson. Story Fragment Stitching: The Case of the Story of Moses. In Proceedings of the first AI4Narratives Workshop, Yokohama, Japan, January 2021

Mohammad Aldawsari, Adrian Perez, Deya Banisakher, and Mark A. Finlayson. Distinguishing Between Foreground and Background Events in News. In Proceedings of the 28th Conference on Computational Linguistics (COLING 2020), online, December 2020