Event Extraction

Event extraction tries to obtain structured representation of events from unstructured natural languages, so as to help answering the "**5W1H**" questions, including "**who, when, where, what, why**" and "**how**"

What is an event ?
An event is a specific occurrence of something that happens in a certain time and a certain place involving one or more participants, which can frequently be described as a **change of state**.*

Goal of event extraction:
Detect event instance(s) in texts, and if existing, identify the event type as well as all of its participants and attributes

Challenges :
It's a challenging task, as events are with different structures and components; while natural languages are often with semantic ambiguities and discourse styles.

***ACE (Automatic Content Extraction) English Annotation Guidelines for Events, Linguistic Data Consortium, Philadelphia, PA, USA, 2005**

Public Evaluation Programs

1. **MUC** (Message Understanding Conference) - by DARPA (1987 to 1997)

2. **TDT** (Topic Detection and Tracking) - by (DARPA + CMU) - NIST developed evaluation system for TDT (1997)

3. **ACE** (Automatic Content Extraction) - by NIST → later converted to TAC (2009 - Text Analysis Conference) (1999)

4. **DEFT** (Deep Exploration and Filtering of Text) - ERE (Entities, Relations, Events Standard for text annotation)

5. **KBP** - Knowledge Base Population

6. **KBP + TAC**

Types of Event Extraction

Closed Domain EE	Open Domain EE
Closed-domain EE uses **predefined event schema** to discover and extract desired events of particular type from text ACE - Ahn Event Structure: 1. **Event mention** : trigger detection 2. **Event Trigger** : trigger type identification 3. **Event Argument** : argument detection 4. **Argument Role** : argument role identification	Without predefined event schemas, open-domain event extraction aims at detecting events from texts and in most cases, also clustering similar events via extracted event keywords. Structure: 1. Event Detection 2. Event Clustering

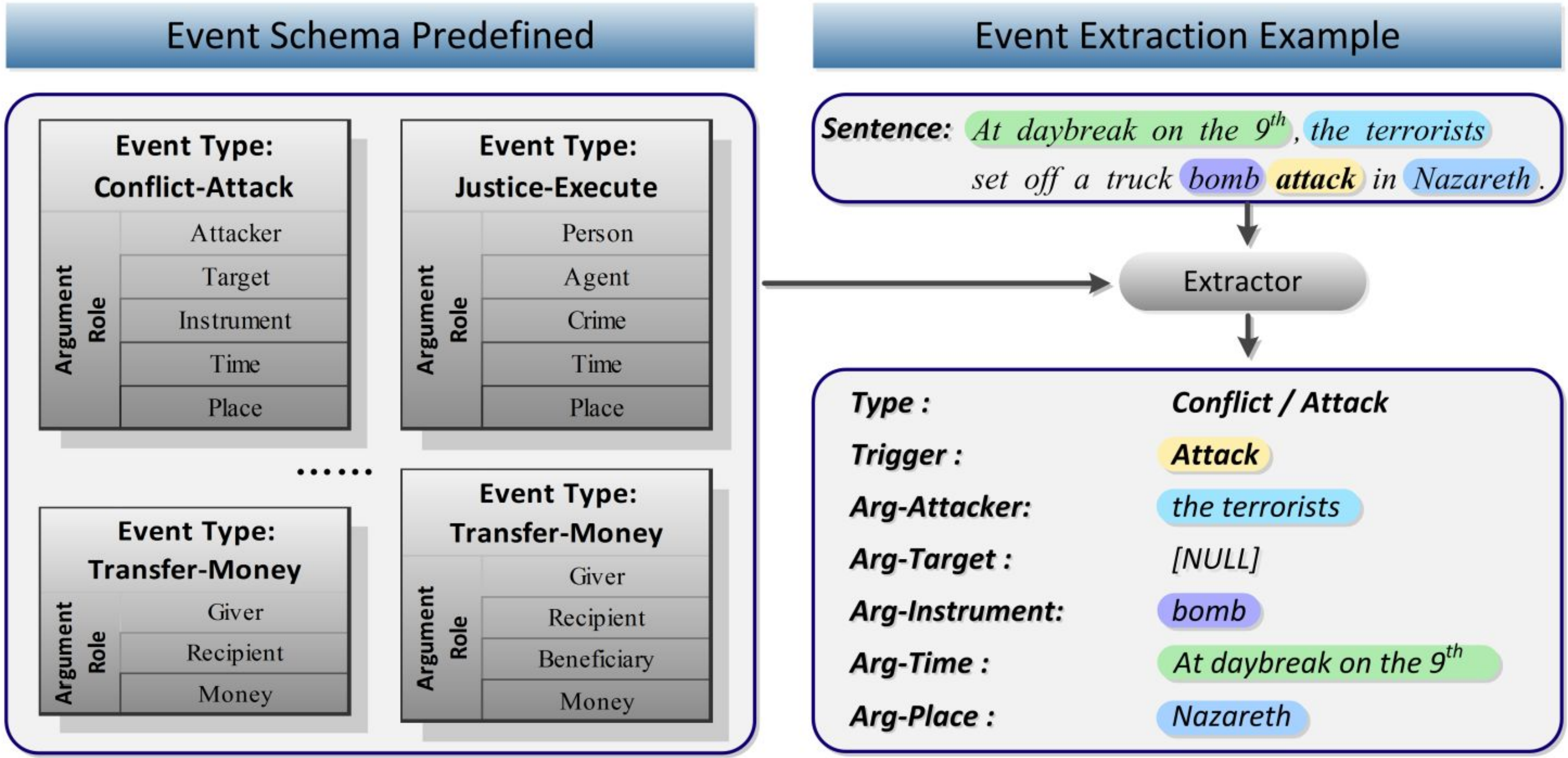


FIGURE 1. Illustration of closed-domain event extraction. The left part illustrates some predefined event schemas in ACE 2005; While the right part illustrates the extraction results of four subtasks for trigger detection, event type identification, argument detection and argument role identification.

Open Domain EE Example

The TDT public evaluation program aims at automatically spotting previously unreported events

- **Story segmentation:** detecting the boundaries of a story from news articles.
- **First story detection:** detecting the story that discuss a new topic in the stream of news.
- **Topic detection:** grouping the stories based on the topics they discuss
- **Topic tracking:** detecting stories that discuss a previously known topic.
- **Story link detection:** deciding whether a pair of stories discuss the same topic.

The first two tasks mainly focus on event detection; and the rest three tasks are for event clustering

Corpus Details

CORPUS	CLASSES	DATA SIZE	TYPE
ACE 2005	8 event types 33 subtypes 36 argument roles	(English, chinese, Arabic) 6000 labeled events	Closed domain
TAC-KBP 2015 (LDC)	9 event types 38 subtypes 3 cat (actual, generic, others)	(English) 158 Documents Train 202 Documents Test + Val	Closed Domain
TDT [TDT1 - TDT5]	100s - topics YES, BRIEF NO tags	English and Chinese (Mandarin) Including both text and speech in both	Open Domain
Domain Specific	BioNLP-ST, TERQAS, TimeBank, MUC		

*About 60% event types have less than 100 instances, and 3 event types have even fewer than 10 instances.

*ACE 2005 and observed that 57% of the trigger words are ambiguous.

TABLE 2: Summary statistics for the datasets. (Doc denotes the number of documents in dataset, Sen denotes the number of sentences in dataset).

Datasets	Doc	Sen	Event Type	Language	Related Papers
MUC-4	1700	-	5	-	[115]
Google	11,909	-	30	English	[153]
Twitter	1,000	-	20	English	[153]
NO.ANN, NO.POS, NO.NEG (DCFEE)	2,976	-	4	Chinese	[91]
ChFinAnn (Doc2EDAG)	32,040	-	5	Chinese	[83]
ACE 2005	599	18,117	33	Multi-language	[33], [42], [75], [80], [84], [89], [95], [143], [154]
TAC KBP 2015	360	12,976	38	English	[14], [78]
TAC KBP 2016	500	9,042	18	Multi-language	[86]
Rich ERE	50			English	[155]
FSED	-	70,852	100	English	[156]
GNBusiness	12,985	1,450,336	-	English	[17]
FSD	-	2,453	20	English	[153]
FBI dataset	-	-	3	English	[157]
RAMS	3,993	-	139	English	[116]
WIKIEVENTS	246	6,132	-	English	[120]
MAVEN	4,480	49,873	168	English	[22] [71] [158]

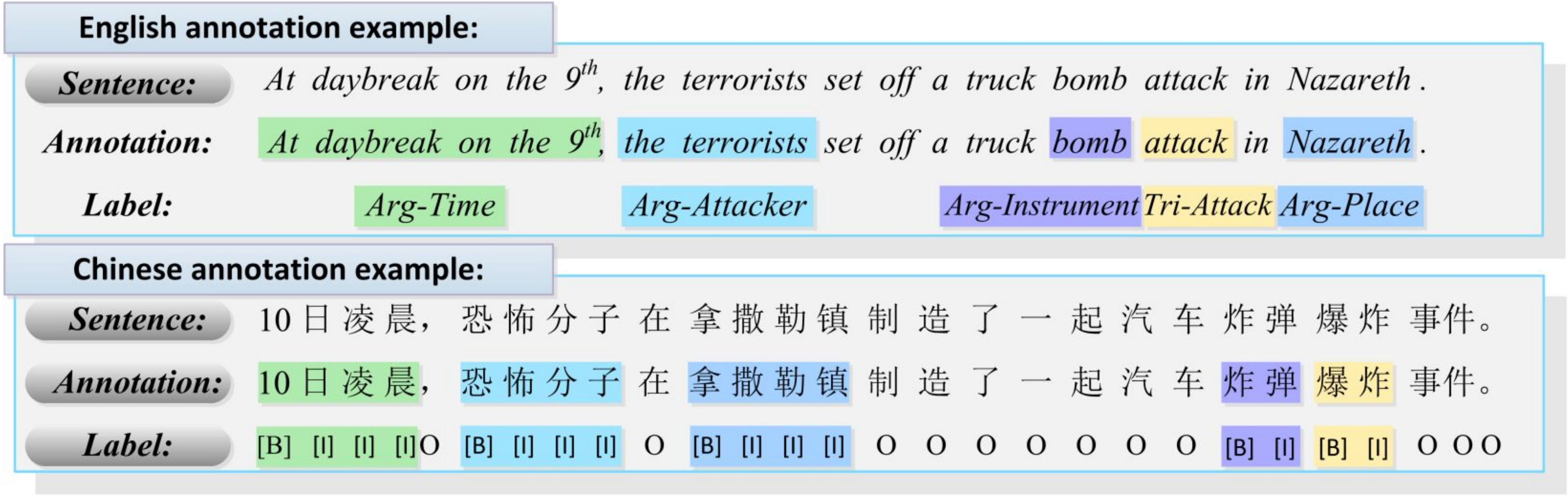
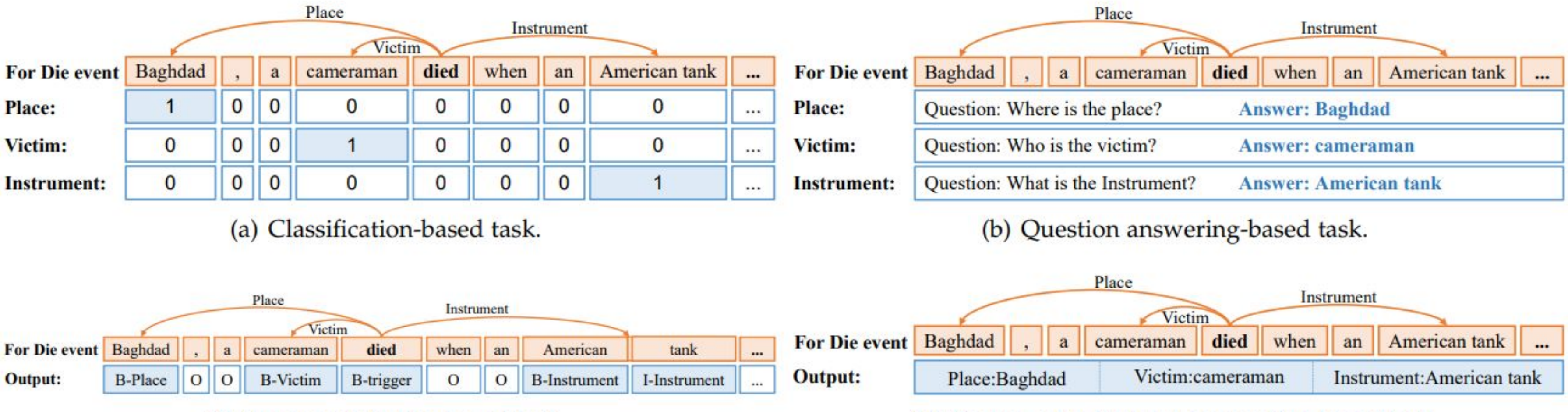


FIGURE 2. Examples of event annotation. In Chinese, events are annotated with the character BIO label (begin/intermediate/other).

Event Extraction Manner

1. Classification-based Task
2. Machine Reading Comprehension-based Task
3. Sequence labeling-based Task
4. Sequence-to-structure Generation-based Task



(a) Classification-based task.

(b) Question answering-based task.

(c) Sequence labeling-based task.

(d) Sequence-to-structure generation-based task.

Taxonomy of Methodologies

1. Event Extraction Based On Pattern Matching
2. Event Extraction Based On Machine Learning
3. Event Extraction Based On Deep Learning
4. Event Extraction Based On Semi-Supervised Learning
5. Event Extraction Based On Unsupervised Learning

Event Extraction Based On Pattern Matching

First constructs specific event templates, and then performs template matching to extract an event with a single argument from text.
In the online extraction phase, an event as well as its argument are extracted if the they match a predefined template.

1. **Manual Pattern Construction (Mostly application Specific)**
   Ex : AutoSlog 1993 for terrorist events [Riloff et al.]
   A small set of linguistic patterns and a manually annotated corpus required.
   The linguistic patterns are used to automatically establish event patterns from manually annotated corpus; While the event patterns are used for event extraction.

2. **Automatic Pattern Construction**
   Apply weakly supervised method or bootstrapping method to obtain more patterns automatically, using only a few pre-classified training corpus or seed patterns.(ML)
   Ex: AutoSlog-TS, ExDisco

Event Extraction Based On Machine Learning

1. **Pipeline Classification Model**

   The pipeline classification normally trains a set of independent classifiers each for one subtask; Yet the output of one classifier can also serve as a part of input to its successive classifier. (SVM, ME)

   local features within sentences and global features within documents:

   a. Sentence Level EE

   b. Document Level EE

2. **Joint Classification Model**

   error propagation problem

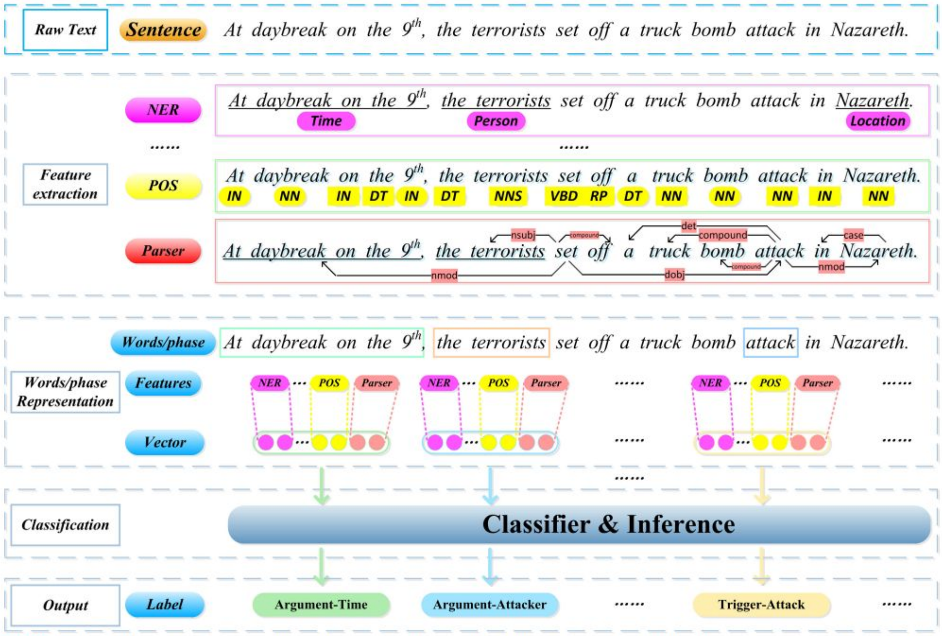


FIGURE 4. Illustration of event extraction based on machine learning. Word/phase features are obtained from execution feature engineering and then input to classifiers for event extraction to output trigger and arguments.

Text Feature for Learning Models

Lexical Features

1. Full word
2. Lowercase Word
3. Proximity Word
4. Lemmatized Word
5. POS tag

Syntactic Features

1. Label of Dependency Path
2. Dependency Words
3. Depth of Candidate word in dependency tree

Semantic Features

1. Synonyms in linguistic dictionary
2. Event and entity type features

Each feature can be represented as a binary vector based on the word-of-bag model

Challenges : Such features are often each with a one-hot representation, which not only suffers from the **data sparsity** problem but also complicates the feature selection for classification model training

Event Extraction Based On Deep Learning

The general process is to build a neural network that **takes word embeddings** as input and outputs a classification result for each word, namely, classifying **whether a word is an event trigger** (or an event argument), and if so, its event type (or argument role)

1. **Convolutional Neural Networks**
   network input : is a concatenation of the word embedding, its position embedding and entity type embedding (**pipeline manner**)

2. **Recurrent Neural Networks**
   CNN cannot well capture some potential interdependencies in between **distant words** to exploit a sentence as a whole for jointly extracting trigger and arguments.
   The recurrent neural network (RNN) structure, which consists of a series of connected neurons, can effectively make use of such sequential inputs. RNN structure also outputs a sequence of words, each of which could be predicted as trigger or argument, whereby the two tasks of joint trigger detection and argument identification can be jointly executed.

Event Extraction Based On Deep Learning

1. Graph Neural Networks

First construct a graph for words in text **abstract meaning representation** (AMR), which can normalize many lexical and syntactic variations in text and output a **directed acyclic graph** to capture the notions of "who did what to whom" in text
The event extraction task as a **subgraph identification** problem.

2. Hybrid Neural Network Models

a. Many researchers have proposed hybrid neural network models which combine different neural networks
b. GAN

3. Attention Mechanism

Argument words to a trigger should receive **more attention** than other words. To this end, they first constructed gold attention vectors to encode only annotated argument words and its contextual words for each annotated trigger. Furthermore, they designed two contextual attention vectors for each word:
One is based on its contextual words; and another is based on its contextual entity type encoded in a transformed entity type space. The two attention vectors are then concatenated and trained together with the event detector to minimize a weighted loss of both event detection and attention discrepancy.

Event Extraction Based On Semi-Supervised Learning

First automatically produce more training data, and then use mixture data containing both original original data and newly generated ones for model training.

1. **Joint Data Expansion and Model Training**
    a. Bootstrapping
    b. Predict and include new training data with High Confidence labels
    c. Trigger-based latent instances (If trigger in one then trigger in others)
    d. Zero-shot transfer learning

2. **Data Expansion from Knowledge bases**
    Exploiting knowledge bases as they store a large amount of structured information, such as FrameNet, Freebase, Wikipedia, WordNet.

3. **Data Expansion from multi language data**
    a. Google Translate to eliminate the language gap between Chinese and English, and produced uniform text representations with bilingual word features
    b. Bootstrap event extraction via exploiting cross-lingual data, i.e. cross-lingual bootstrapping, without using machine translation

Event Extraction Based On Unsupervised Learning

Focus on open-domain event extraction tasks, like detecting trigger and arguments based on word distributional representations

1. **Event Mention Detection and Tracking**
   Sentences are firstly converted into vector representations, and the vector distance is computed to measure the similarity to some topic for event detection.

2. **Event Extraction and Clustering**
   Regard the verb of a sentence as an event trigger.
   Used verbs in sentences as event triggers and identified event arguments by utilizing the dependency paths between triggers and named entities, time expressions, sentence subjects and sentence objects.
   Based on the extracted triggers and arguments, event instances can be clustered into different event groups each with a latent yet distinguished topic.

3. **Event Extraction from Social Media**
   Working on Twitter Data

TABLE 4. **Performance comparison of English event extraction on the ACE 2005 dataset.**

System	Approach	Results-F1			
		Trigger	Event Type	Argument	Argument Role
David Ahn (2006) [27]	Machine Learning	62.6%	60.1%	57.3%	-
Ji & Grishman (2008) [102]	Machine Learning	-	67.3%	46.2%	42.6%
Liao & Grishman (2010) [103]	Machine Learning	-	68.8%	50.3%	44.6%
Das et al. (2010) [116]	Machine Learning	-	60.0%	-	41.0%
Hong et al. (2011) [106]	Machine Learning	-	68.3%	53.2%	48.4%
Liao et al. (2011) [107]	Machine Learning	-	61.7%	39.1%	35.5%
Li et al. (2013) [113]	Machine Learning	70.4%	67.5%	56.8%	52.7%
Li et al. (2014) [128]	Machine Learning	-	65.2%	-	46.8%
Cao et al. (2015) [79]	Pattern Matching	-	70.4%	-	-
Chen et al. (2015) [149]	Convolutional Neural Networks	73.5%	69.1%	59.1%	53.5%
Nguyen & Grishman (2015) [141]	Convolutional Neural Networks	-	69.0%	-	-
Judea & Strube (2016) [129]	Machine Learning	66.5%	63.7%	53.1%	41.8%
Sha et al. (2016) [96]	Machine Learning	-	68.9%	61.2%	53.8%
Liu et al. (2016) [105]	Machine Learning	-	69.4%	-	-
Yang & Mitchell (2016) [130]	Machine Learning	71.0%	68.7%	50.6%	48.4%
Zhang et al. (2016) [151]	Convolutional Neural Networks	74.8%	69.1%	58.6%	53.1%
Nguyen et al. (2016) [161]	Recurrent Neural Networks	71.9%	69.3%	62.8%	55.4%
Chen et al. (2016) [163]	Recurrent Neural Networks	72.2%	68.9%	60.0%	54.1%
Feng et al. (2016) [185]	Hybrid Neural Networks	75.9%	73.4%	-	-
Liu et al. (2017) [142]	Artificial Neural Networks	72.3%	69.6%	-	-
Liu et al. (2017) [197]	Artificial Neural Networks	-	71.7%	-	-

Yang & Mitchell (2016) [130]	Machine Learning	71.0%	68.7%	50.6%	48.4%
Zhang et al. (2016) [151]	Convolutional Neural Networks	74.8%	69.1%	58.6%	53.1%
Nguyen et al. (2016) [161]	Recurrent Neural Networks	71.9%	69.3%	62.8%	55.4%
Chen et al. (2016) [163]	Recurrent Neural Networks	72.2%	68.9%	60.0%	54.1%
Feng et al. (2016) [185]	Hybrid Neural Networks	75.9%	73.4%	-	-
Liu et al. (2017) [142]	Artificial Neural Networks	72.3%	69.6%	-	-
Liu et al. (2017) [197]	Artificial Neural Networks	-	71.7%	-	-
Duan et al. (2017) [170]	Recurrent Neural Networks	-	70.5%	-	-
Sha et al. (2018) [166]	Recurrent Neural Networks	-	71.9%	67.7%	58.7%
Wu et al. (2018) [198]	Recurrent Neural Networks	73.4%	71.6%	-	-
Zhang et al. (2018) [200]	Recurrent Neural Networks	76.1%	73.9%	-	-
Ding et al. (2018) [201]	Recurrent Neural Networks	74.9%	71.2%	64.8%	56.6%
Zhao et al. (2018) [203]	Recurrent Neural Networks	-	74.0%	-	-
Chen et al. (2018) [204]	Recurrent Neural Networks	-	73.3%	-	-
Li et al. (2018) [207]	Recurrent Neural Networks	-	75.6%	-	-
Liu et al. (2018) [206]	Recurrent Neural Networks	74.1%	72.4%	-	-
Liu et al. (2018) [179]	Graph Neural Networks	75.9%	73.7%	68.4%	60.3%
Nguyen & Grishman (2018) [180]	Graph Neural Networks	-	73.1%	-	-
Liu et al. (2018) [189]	Hybrid Neural Networks	65.4%	-	-	-
Hong et al. (2018) [191]	Hybrid Neural Networks	77.0%	73.0%	-	-
Nguyen & Nguyen (2019) [172]	Recurrent Neural Networks	72.5%	69.8%	59.9%	52.1%
Zhang et al. (2019) [173]	Recurrent Neural Networks	72.9%	71.6%	-	-
Zhang et al. (2019) [193]	Hybrid Neural Networks	74.6%	72.9%	67.9%	59.7%
Liu et al. (2019) [192]	Hybrid Neural Networks	-	74.8%	-	-