IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 4, 2019, accepted November 26, 2019, date of publication November 29, 2019,
date of current version December 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2956831

A Survey of Event Extraction From Text

WEI XIANG AND BANG WANG
School of Electronic Information and Communications, Huazhong University of Science and Technology (HUST), Wuhan 430074, China
Corresponding author: Bang Wang (wangbang@hust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61771209.

ABSTRACT Numerous important events happen everyday and everywhere but are reported in different media sources with different narrative styles. How to detect whether real-world events have been reported in articles and posts is one of the main tasks of event extraction. Other tasks include extracting event arguments and identifying their roles, as well as clustering and tracking similar events from different texts. As one of the most important research themes in natural language processing and understanding, event extraction has a wide range of applications in diverse domains and has been intensively researched for decades. This article provides a comprehensive yet up-to-date survey for event extraction from text. We not only summarize the task definitions, data sources and performance evaluations for event extraction, but also provide a taxonomy for its solution approaches. In each solution group, we provide detailed analysis for the most representative methods, especially their origins, basics, strengths and weaknesses. Last, we also present our envisions about future research directions.

INDEX TERMS Event extraction, event extraction tasks, event corpus, natural language processing.

I. INTRODUCTION

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 [1]. The goal of event extraction is to detect event instance(s) in texts, and if existing, identify the event type as well as all of its participants and attributes. Although different event types may be defined with different arguments, a simple summary of event extraction is 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'', of real-world events from numerous sources of texts, like news articles, social media posts and etc.

As an important task of information retrieval in the *natural language processing* (NLP), event extraction has lots of applications in diverse domains. For example, the structured events can be directly used to expand knowledge bases upon which further logical reasoning and inference can be made [2], [3]. Event detection and monitoring have long been the focus of public affair management for governments, as timely knowing the outbursts and evolutions of popular social events help the authorities to respond promptly [4]–[8]. In the business and financial domain, event extraction can also help companies quickly discovering market responses of their products and inferencing signals for risks analysis and trading suggestions [9]–[11]. In biomedical domain, event extraction can be used to identify the alterations in the state of a biomolecule (e.g. gene and protein) or interactions between two or more biomolecule, which is described in natural language in the scientific literature for the understanding of physiological and pathogenesis mechanisms [12]. In short, many domains can benefit from the advancements of event extraction techniques and systems.

Despite its promising applications, event extraction is still a rather challenging task, as events are with different structures and components; while natural languages are often with semantic ambiguities and discourse styles. Furthermore, event extraction is also closely related to other NLP tasks, like *named entity recognition* (NER), *part-of-speech* (POS) tagging, *syntactic parsing* and etc., which can either boost event extraction or reversely impact on its performance, depending on how these tasks can perform and how to exploit their outputs. To prompt the developments and applications of event extraction, many public evaluation programs have been conducted to provide task definitions, annotated corpora as well as open contests to promote information extraction research like event extraction, which have also been attracting many talents to contribute novel algorithms, techniques and systems. We next briefly introduce these well-known programs.

A. PUBLIC EVALUATION PROGRAMS

The *Message Understanding Conference* (MUC) [13], [14] has been generally recognized as the first public evaluation

The associate editor coordinating the review of this manuscript and approving it for publication was Nikhil Padhi.

program for information extraction, which was organized and sponsored by the Defense Advanced Research Projects Agency (DARPA).[1] It had held for seven times from 1987 to 1997. The MUC aims at extracting information from unstructured text and populating into a slot-value structure in predefined schemas. Some common slots include entities, attributes, entity relations, events and etc. In 1997, the DARPA, Carnegie Mellon University, Dragon Systems, and the University of Massachusetts at Amherst co-founded another public evaluation program, called *Topic Detection and Tracking* (TDT), to promote finding and following new events in a stream of broadcast news articles [15], [16]. Later on, the National Institute of Standards and Technology (NIST)[2] established a complete evaluation system for the TDT program.

The *Automatic Content Extraction* (ACE) is the most influential public evaluation programs up to now, which was proposed by NIST in 1999 and later on incorporated into a new public evaluation program *Text Analysis Conference* (TAC) in 2009. From 2000 to 2004, the ACE was devoted to entity and relation detection and tracking, and the event extraction task had been appended into the ACE since 2005 [17]. Following the ACE, the *Deep Exploration and Filtering of Text* (DEFT) program of DARPA proposed the *Entities, Relations, Events* (ERE) standard for text annotation and information extraction. The Light ERE was defined as a simplified version of the ACE annotation in order to rapidly produce consistently labeled data. Subsequently, the Light ERE has been extended to a more complex Rich ERE specification [18].

Moreover, event extraction has served as the mainstream task of the *Knowledge Base Population* (KBP) public evaluation program that had been held for four times since 2014 up to now. Now the KBP has integrated with the TAC, with the aims of extracting information from a large text corpus to complete deficient elements for knowledge bases [19]. In addition, there are also some other event public evaluation programs for event extraction in specific domains, such as the BioNLP in the biomedical domain [20], the TimeBANK for extracting temporal information of events [21].

B. SUMMARY OF THIS SURVEY

This article provides an up-to-date survey for event extraction from text. We note that there are some related survey articles on this task, yet each with a particular focus for specific application domain. Hogenboom et al. [22], [23] reviewed the text mining techniques for event extraction in various decision support systems. Vanegas *et al.* [12] mainly reviewed the biomolecular event extraction; While Zhang *et al.* [24] mainly focused on open domain event extraction; Some have also focused on event extraction from social media, especially from Twitter [25], [26].

[1]https://www.darpa.mil
[2]https://www.nist.gov

Compared with the aforementioned articles, we try to provide a more comprehensive survey and systematic technique taxonomy for event extraction from text, not only providing its task definitions, data sources and performance evaluations, but also categorizing the main approaches from the viewpoint of its whole development history. Furthermore, we present and analyze the most representative methods in each technique class, especially their origins, basics, advantages and disadvantages. We also discuss the promising directions for future research of event extraction.

The rest of the article is organized as follows: Section II introduces the main task definitions in both closed-domain and open-domain event extraction; While the mostly used corpora are introduced in Section III. We categorize the main technique approaches into 5 groups, including those for closed-domain event extraction, the earlier approaches of event pattern matching in Section IV, machine learning methods in Section V, deep learning models in Section VI, semi-supervised learning schemes in Section VII, and for open-domain event extraction, the unsupervised learning approaches in Section VIII. We also compare the extraction performance for those algorithms experimented on the ACE corpus in Section IX. Finally, Section X concludes the survey with some discussions about future research directions.

II. EVENT EXTRACTION TASKS

Event extraction aims at detecting the existence of an event reported in text, and if existing, discovering event-related information from the text, such as the "5W1H" about an event (i.e., *who*, *when*, *where*, *what*, *why* and *how*). Sometimes, particular event structures are predefined, which includes not only event types but also event arguments' roles. Event extraction needs not only to detect an event but also extracting the corresponding characters/words/prases to fill in the given event structure, so as to output a structured form of event. This is normally called *closed-domain event extraction*, as different domains may need different event structures. On the other hand, the task of *open-domain event extraction* does not assume predefined event structures, and the main task is to detect the existence of an event in text. In many cases, it also extracts keywords about an event and clusters similar events.

A. CLOSED-DOMAIN EVENT EXTRACTION

Closed-domain event extraction uses predefined *event schema* to discover and extract desired events of particular type from text. An event schema contains several event types and their corresponding event structures. We use the ACE [1] terminologies to introduce an event structure as follows:

- **Event mention:** a phrase or sentence describing an event, including a trigger and several arguments.
- **Event trigger:** the main word that most clearly expresses an event occurrence, typically a verb or a noun.

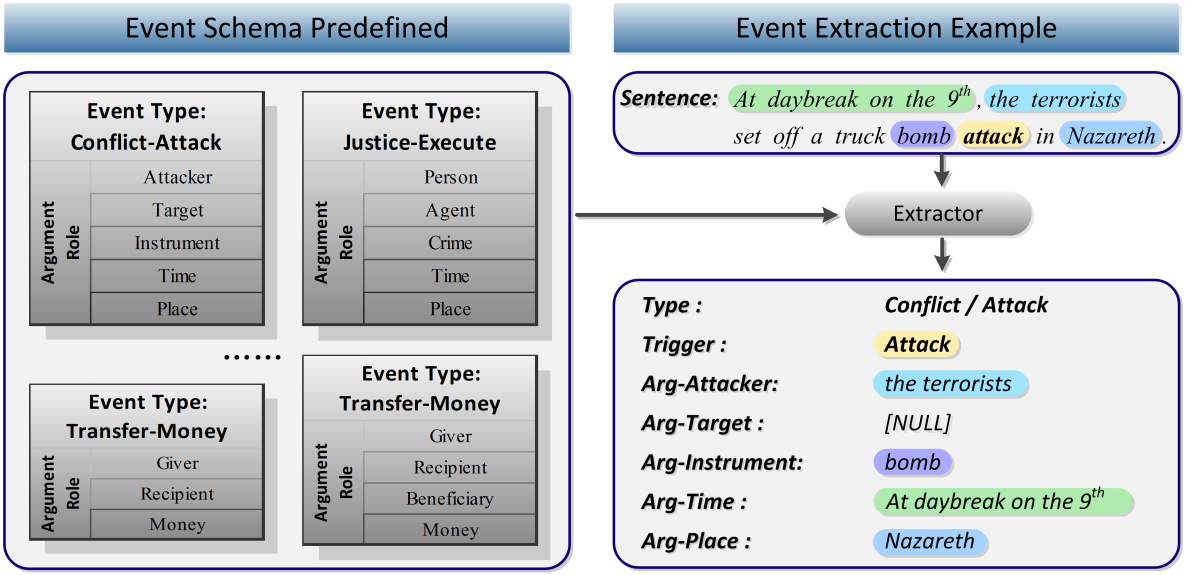


FIGURE 1. Illustration of closed-domain event extraction. The left part illustrates some predefined event schemas in ACE 2005; While the right part illustrates the extraction results of four subtasks for trigger detection, event type identification, argument detection and argument role identification.

- **Event argument:** an entity mention, temporal expression or value that serves as a participant or attribute with a specific role in an event.
- **Argument role:** the relationship between an argument to the event in which it participants.

Note that a same event may be mentioned multiple times in different sentences or documents. How to distinguish multiple event mentions of a same event belongs to another crucial NLP task, called *event co-reference*. In this paper we do not consider event mention co-reference resolution.

Ahn [27] first proposed to divide the ACE event extraction task into four subtasks: trigger detection, event/trigger type identification, event argument detection, and argument role identification. For example, consider the following sentence:

Sentence 1: At daybreak on the 9th, the terrorists set off a truck bomb attack in Nazareth.

There exists an event of type "Conflict/Attack" in Sentence 1. An event extractor should discover such an event and identify its type by detecting the trigger word "attack" in the sentence and classifying it to the event type of "Conflict/Attack". It next should extract all arguments related to this event type from the text and identify their respective roles according to the predefined event structure.

Fig. 1 illustrates the closed-domain extraction for structured events. 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.

Following similar definitions by ACE [1], many other event types and structures have been defined and adopted in the field, like those defined by ERE and TAC-KBP [17]–[19]. Besides organizations, individual researchers have also defined event types and structures for specific domains. Petroni *et al.* [28] defined structures for breaking events, including 7 event types like "Floods", "Storms", "Fires" and etc., as well as their "5W1H" attributes, so as to extract breaking events from news reports and social media. Yang *et al.* [29] focused on extracting events in the financial domain to help predicting the stock market, investment decision support and etc. They defined 9 financial event types, like "Equity Pledge", "Equity Freeze" and etc., as well as their corresponding arguments with different roles. Han *et al.* [30] defined a taxonomy of business events containing 8 event types with 16 subtypes with their corresponding arguments, like tense, time, result, entity etc.

B. OPEN-DOMAIN EVENT EXTRACTION

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. Event keywords refer to those words/phrases mostly describing an event, and sometimes keywords are further divided into triggers and arguments.

The TDT public evaluation program aims at automatically spotting previously unreported events or following the progress of the previously spotted events from news articles [15]. Besides events, the TDT also defines the *story* as a segment of news article describing a specific event, and *topic* as a set of events in articles yet strongly related to some real-world topic. Based on such definitions, it defines the following tasks:

- **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. While the relation between the five tasks is evident, each requires a distinct evaluation process and encourages different approaches to address the particular problem.

Besides the TDT tasks, many other researches have also been conducted for detecting and clustering open-domain events from news articles [6], [31]–[34]. For example, the Joint Research Centre of the European Commission investigated extracting violent events with keywords, like killed, injured, kidnapped and etc., from online news for global crisis surveillance [6], [31]. Yu and Wu [33] aggregated news articles about a same event into a topic-centered set; While Liu *et al.* [34] clustered news articles according to daily significant events about politics, economics, societies, sports, entertainment and etc.

Some work have focused on sentence-level event detection and clustering [35]–[37]. For example, Naughton *et al.* [35] grouped sentences in news articles that refer to the same event, where non-event sentences are removed before initiating the clustering process. They used a set of news stories collected from different sources describing events related to the Iraq war. Moreover, they designed clustering labels, such as terrorist attack, bombing, shooting, air attack and etc., when grouping event-related sentences. Besides event detection and clustering, Wang *et al.* [36] also proposed to extract keywords for each event, like the type, location, time and people about an event.

Other than newswire articles, many online social media, such as Twitter and Facebook etc., provide abundant and timely information about diverse types of events. Detecting and extracting events from social media have also been becoming an important task recently [28], [38]–[43]. It is worth noting that as posts in social networks are kind of unofficial texts with lots of abbreviations, misspellings and grammar errors, how to extract events from such online posts faces more challenges than extracting events from news articles.

Although detecting and clustering events are the main tasks in open-domain event extraction, some researchers have also proposed to further construct event schemas from the clustered event-related sentences and documents by assigning each event cluster an event type label as well as one or more event attribute labels [44]–[52]. Notice that such cluster labels might be better explained as a kind of semantic synthesis from the keywords of each cluster, other than the predefined ones with clear structure as that in the closed-domain event extraction.

III. EVENT EXTRACTION CORPUS

This section primarily introduce the corpus resource of event extraction tasks. Generally, public evaluation programs provide several corpus for task evaluation of event extraction. The corpus are manually annotated by public evaluation programs according to the task definition, which is also used for model training and verification in machine learning methods.Sample annotations are completed by professionals or experts with domain knowledge and annotated samples can be regarded as with *ground truth* labels. However, as the annotation process is cost-prohibitive, many public corpora are with small size and low coverage.

A. THE ACE EVENT CORPUS

The ACE program [1] provided annotated data and evaluation tools for various extraction tasks, including entity, time, value, relation and event extraction. Entities in ACE fall into 7 types (person, organization, location, geo-political entity, facility, vehicle and weapon) each with a number of subtypes. Furthermore, time is annotated according to the TIMEX2 standard [53], [54], which is a rich specification language for event and temporal expressions in natural language text. Every text sample was dually annotated by two independent annotators, and a senior annotator adjudicated the version discrepancies between them.

Events in the ACE corpus have complex structures and arguments involving entities, times, and values. The ACE 2005 event corpus defined 8 event types and 33 subtypes, each event subtype corresponding to a set of argument roles. There are in total 36 argument roles for all event subtypes. In most of researches based on the ACE corpus, the 33 subtypes of events are often treated separately without further retrieving their hierarchical structures. Table 1 provides these event types and their corresponding subtypes. An example of annotated event sample has been provided in Fig. 2. The ACE 2005 corpus contains in total 599 annotated documents and about 6000 labeled events, including English, Arabic and Chinese events from different media sources like newswire articles, broadcast news and etc. Table 2 provides their source statistics.

TABLE 1. Event types and subtypes in ACE 2005.

SN	Event Type	SN	Event subtype
1	Life	1-5	Be-Born, Marry, Divorce, Injure, Die
2	Movement	6	Transport
3	Contact	7-8	Meet, Phone-write
4	Conflict	9-10	Attack, Demonstrate
5	Business	11-14	Merge-org, Declare-bankruptcy, Start-Org, End-org
6	Transaction	15-16	Transfer-money, Transfer-ownership
7	Persosnnel	17-20	Elect, Start-position, End-position, Nominate
8	Justice	21-33	Arrest-jail, Execute, Pardon, Release-parole, Fine, Convict, Charge-indict, Trial-hearing, Acquite, Sentence, Sue, Extradite, appeal

B. THE TAC-KBP CORPUS

The *event nugget detection* task in TAC-KBP focuses on detecting explicit mentions of event with its types and

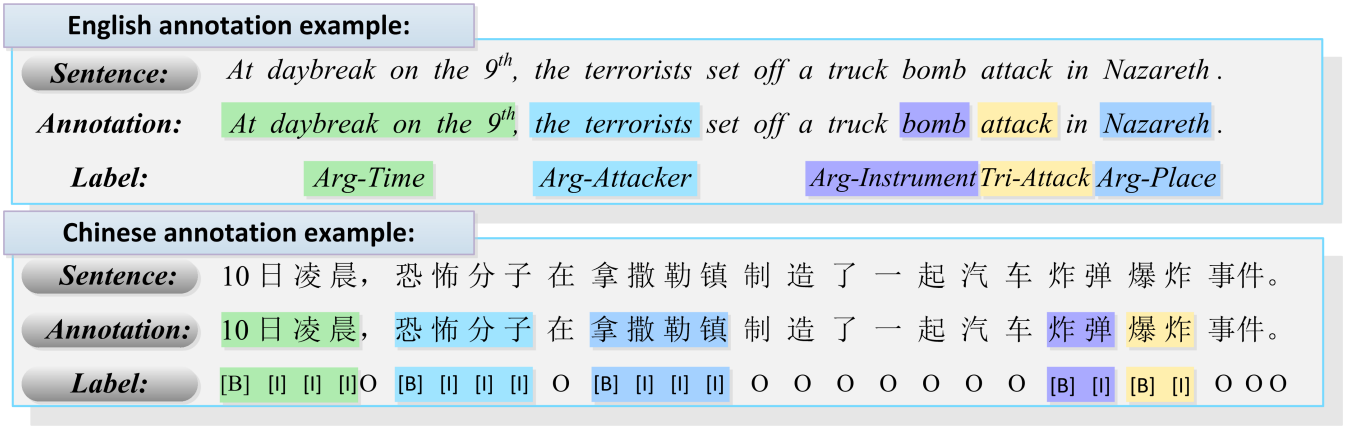


FIGURE 2. Examples of event annotation. In Chinese, events are annotated with the character BIO label (begin/intermediate/other).

TABLE 2. Data sources and data statistics in ACE 2005.

Data sources	Language		
	English	Chinese	Arabic
Newswire (NW)	20%	40%	40%
Broadcast News (BN)	20%	40%	40%
Broadcast Conversation (BC)	15%	0%	0%
Weblog (WL)	15%	20%	20%
Usenet Newsgroups (UN)	15%	0%	0%
Conversational Telephone Speech (CTS)	15%	0%	0%

subtypes as defined in Rich ERE. The TAC-KPB 2015 corpus provided by Linguistic Data Consortium(LDC) includes 158 documents as prior training set and 202 additional documents as test set for the formal evaluation from newswire articles and discussion forum [55]. The event types and subtypes in TAC-KBP (Rich ERE) are defined referring to the ACE corpus, including 9 event types and 38 subtypes. In addition, event mentions must be assigned into one of the three REALIS valus: ACTUAL(actually occurred), GENERIC(without specific time or place), and OTHERS(non-generic events, such as failed events, future events, and conditional statements etc.) [19]. The TAC-KBP 2015 corpus is only for English evaluation, however, Chinese and Spanish have been added in TAC-KBP 2016 for all tasks.

C. THE TDT CORPUS

In open-domain event extraction, the LDC also provides a series of corpus to support TDT research from TDT-1 to TDT-5, including both text and speech in both English and Chinese (Mandarin) [15], [56]. Each TDT corpus contains millions of news stories annotated with hundreds of topics (events) collected from multiple sources like newswire and broadcast articles. Furthermore, all the audio materials are transformed into text intermediaries by LDC. These story-topic tags are assigned a value of *YES*, if the story discusses the target topic, or *BRIEF* if that discussion comprises less than 10% of the story. Otherwise, the (default) tag value is *NO*.

D. OTHER DOMAIN-SPECIFIC CORPORA

Besides the aforementioned well-known corpora, some domain-specific event corpora have also been established and released. The BioNLP Shared Task (BioNLP-ST) is defined for fine-grained biomolecular event extraction from scientific documents in the biomedical domain, which compiled various manually annotated biological corpus including GENIA event corpus, BioInfer corpus, Gene regulation event corpus, GeneReg corpus and PPI corpora [12], [20]. The TERQAS (Time and Event Recognition for Question Answering Systems) workshop has built a corpus, called TimeBANK, which was annotated for events, times, and temporal relations from a wide variety of media sources for breaking news events extraction [21]. Meng *et al.* [57] also annotated breaking news events reported in Chinese, called CEC (Chinese Event Corpus). Other domain-specific corpora include the MUC series corpus for event extraction in the field of military intelligence, Terrorist attacks, Chip technology and financial [13], [14], and Ding *et al.*'s corpus for event extraction in the field of music [58].

IV. EVENT EXTRACTION BASED ON PATTERN MATCHING

The earlier approaches for event extraction are a kind of pattern matching technique, which first constructs some specific event templates, and then performs template matching to extract an event with a single argument from text. As illustrated by Fig. 3, event template can be constructed from raw texts or annotated texts, yet both require profession knowledge. In the online extraction phase, an event as well as its argument are extracted if the they match a predefined template.

A. MANUAL PATTERN CONSTRUCTION

We first review some typical pattern-based event extraction systems, which employ experts with professional knowledge to manually constructing event patterns for different domain

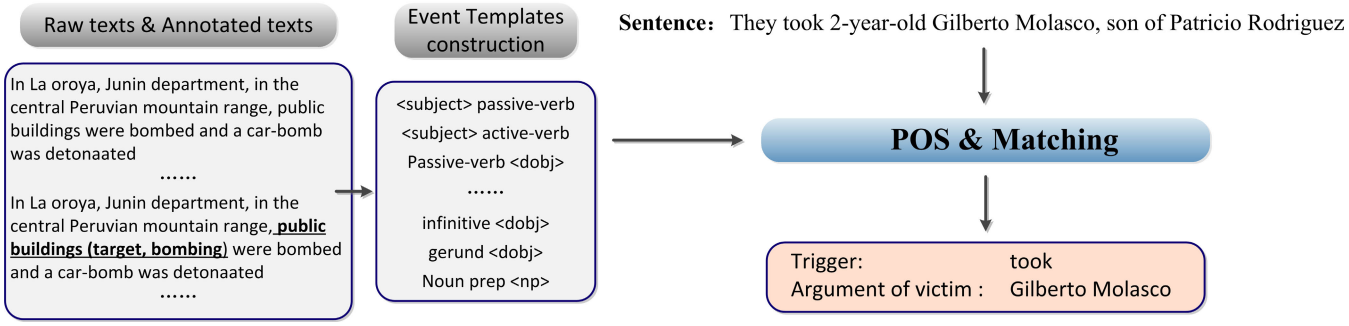


FIGURE 3. Illustration of event pattern construction and event extraction based on pattern matching.

applications. The first pattern-based extraction system might be dated back to the AutoSlog developed in 1993 by Riloff *et al.* [59], which was a domain-specific one for extracting terrorist events. The AutoSlog exploited a small set of linguistic patterns and a manually annotated corpus to obtain event patterns. As presented in Table. 3, in total 13 linguistic patterns were defined, such as "<subject> passive-verb", meaning a phrasal verb in passive form followed by a grammatical element acting as the subject. Note that the linguistic patterns are distinguished from the event patterns in AutoSlog. The linguistic patterns are used to automatically establish event patterns from manually annotated corpus; While the event patterns are used for event extraction. Moreover, the AutoSlog was designed to extract an event with a single event argument, thus their corpus were annotated with a single argument for each event. For example, in the following sentence, the underlined phrase *public buildings* is annotated as an argument of "target" role with an event of "bombing" type.

Sentence 2: In La oroya, Junin department, in the central Peruvian mountain range, ***public buildings*** *(target, bombing) were bombed and a car-bomb was detonated.*

Specifically, the AutoSlog employed a syntactic analyzer CIRCUS [60] to identify the *Part of Speech* (POS) of each sentence in manually annotated corpus, such as subjects, predicates, objects, prepositions etc. Subsequently, It generates a trigger word dictionary and concept nodes, which can be viewed as event patterns, involving event trigger, event type, event argument and argument role. In **Sentence 2**, the phrase of *"public buildings"* is identified as a subject by CIRCUS. Since it is followed by the passive verb "bombed", it matches the pre-defined linguistic pattern "<subject> passive verb". As a result, it adds "bombed" into the trigger word dictionary and generates an event pattern "<target> was bombed" with the bombing type.

In the process of pattern matching (event extraction), the AutoSlog first uses the trigger word dictionary to locate candidate event sentences, and then performs the POS on the candidate sentences by CIRCUS. It next associates the syntax features surrounding the trigger word (i.e. the output of POS using CIRCUS) and the event patterns to extract argument and its role of event. For a predefined an event pattern "took <victim>" with kidnapping type, its trigger word "took" would locate a candidate sentence "*they took 2-year-old Gilberto Molasco, son of Patricio Rodriguez*". The result of POS using CIRCUS further identifies that "Gilberto Molasco" is a dobj (direct object). Finally, combing the POS results and the predefined event pattern, the AutoSlog can extract an event of kidnapping type with its trigger word "took" and argument of victim "Gilberto Molasco".

TABLE 3. The linguistic pattern and example in AutoSlog. The bracketed item shows the syntactic constituent where the string was found (e.g. <dobj> is the direct object and <np> is the noun phrase following a preposition). In the examples, the bracketed item is a slot name that might be associated with the filter (e.g., the subject is a victim). The underlined word is the trigger.

SN	Linguistic Pattern	Example
1	<subject> passive-verb	<victim> was murdered
2	<subject> active-verb	<perpetrator> was bombed
3	<subject> verb infinitive	<perpetrator> attempted to kill
4	<subject> auxiliary noun	<victim> was victim
5	passive-verb <dobj>	killed <victim>
6	active-verb <dobj>	bombed <target>
7	infinitive <dobj>	to kill <victim>
8	verb infinitive <dobj>	threatened to attack <target>
9	gerund <dobj>	killing <victim>
10	noun auxiliary <dobj>	fatality was <victim>
11	noun prep <np>	bomb against <target>
12	active-verb <np>	killed with <instrument>
13	passive-verb <np>	was aimed at <target>

Inspired by the AutoSlog system, many pattern-based event extraction systems have been developed for different application domains, including biomedical event extraction [61]–[64], financial event extraction [65], [66] and etc. For example, Cohen *et al.* [61] extracted biomedical events via the biomedical ontology analysis by exploiting the OpenDMAP semantic parser [67], which provides various high-quality ontological templates for biomedical concepts and their properties. Casillas *et al.* [62] applied the Kybots (Knowledge Yielding Robots) developed by the KYOTO project[3] to extract biomedical events. The Kybots system

[3]http://www.kyoto-project.eu

follows a rule-based approach to manually build event patterns. In the financial domain, Borsje *et al.* [65] proposed a semi-automatic financial event extraction method based on lexico-semantic patterns from news feeds. Arendarenko and Kakkonen [66] developed an ontology-based event extraction system, called BEECON, to extract business events from online news. Since constructing event patterns often requires professional knowledge, these pattern-based extraction systems are normally designed for specific application domain. For applications involving multiple domains, Cao *et al.* [68] proposed to built event patterns by combining ACE corpus with other expert-defined patterns, such as the TABARI corpus,[4] to produce more event patterns.

Besides many systems designed for English event extraction, some pattern-based extraction systems have been designed for other languages [69], [70]. Tran *et al.* [69] developed a real-time extraction framework, called VnLoc, by combining lexico-semantic rules with machine learning to extract events from Vietnamese news. Saroj *et al.* [70] designed a rule-based event extraction system from newswires and social media text for Indian language. Valenzuela-Escárcega *et al.* [71] observed that there is not a standard language to express patterns, which could hindered the development of the pattern-based event extraction task. They proposed a domain-independent, rule-based framework and designed rapid development environment for event extraction to reduce the cost for newcomers.

Although pattern-based event extraction can achieve high extraction accuracy, yet the pattern construction suffers the scalability problem in that the patterns are often dependent on the application domain. To increase scalability, Kim *et al.* [72] designed the PALKA (Parallel Automatic Linguistic Knowledge Acquisition) system to automatically acquire event patterns from annotated corpus. They defined a specialized representation for event patterns, called FP-structures (Frame-Phrasal and pattern structure). In PALKA, event patterns were constructed in the form of FP-structures, and further tuned through the generalization of semantic constraints. Furthermore, the FP-structures extended the single event argument pattern to event pattern with multiple arguments, i.e, an event may contain multiple arguments. For example, the ''bombing'' event in **Sentence 2**, in addition to the ''target'' argument, it may have multiple potential event arguments, such as ''agent'', ''patient'', ''instrument'' and ''effect''. Furthermore, the PALKA converts multiple clauses in one sentence into a multi-sentence form for multiple event extraction. In addition, Aone and Ramos-Santacruz [73] developed a large-scale event extraction system, called REES, which can extract up to 100 event types by establishing event and relation ontology schemas for generating new event patterns, yet the new patters are still subject to manual validation through the graphical interface of their system.

B. AUTOMATIC PATTERN CONSTRUCTION

For event patterns manually constructed by experts with professional knowledge, as they are well-defined with high quality, so event extraction based on pattern matching can often achieve high accuracy for domain-specific applications. However, manually constructing event patterns is rather time-consuming and labor-intensive, leading to not only the scalability problem for producing large pattern databases but also the adaptivity problem when applying event patterns in other domains. Some researchers have proposed to apply weakly supervised method or bootstrapping method to obtain more patterns automatically, using only a few pre-classified training corpus or seed patterns.

Riloff *et al.* [74] developed the AutoSlog-TS event extraction system to enable automatic pattern construction, which was an extension of their previous AutoSlog system. In particular, base on the thirteen linguistic patterns defined in the AutoSlog system, it applies a syntactic analyzer CIRCUS to obtain new event patterns from untagged corpus. The following example illustrates its new pattern construction process.

Sentence 3: World trade center was bombed by terrorists.

Via syntactic analysis, we can get the subject ''World trade center'', the verb phrase ''was bombed'' and the prepositional phrase ''by terrorists''. Combining the ''<subject> passive-verb'' and ''passive-verb prep <np>'' in linguistic patterns, we can get potential event patterns: ''<x> was bombed'' and ''bombed by <y>''. Whether the new event patterns are included is dependent on their scorings according to the statistics in both domain-dependent and domain-independent documents. Their experiments on MUC-4 terrorism dataset[5] validated that the AutoSlog-TS dictionary performs comparably to a hand-crafted dictionary. Despite some manual intervention is still required, the AutoSlog-TS can significantly reduce the workloads to create a large training corpus.

Many other approaches have been proposed to facilitate automatical pattern construction by designing machine learning algorithms to learn new patterns based on a few seed patterns [6], [8], [31], [75]–[84]. For example, the ExDisco proposed by Yangarber *et al.* [75] provided a small set of seed patterns instead of linguistic patterns to obtain potential event patterns. Not only seed patterns, but also seed terms or seed event instances have also been exploited to construct potential event patterns [78], [81]–[84]. The NEXUS system developed by Piskoriski *et al.* [8], [31] learned candidate event patterns via an entropy maximization-based machine learning algorithm from a small set of annotated corpus, yet the candidate patterns are still manually checked and modified to be included into the pattern database. Cao *et al.* [79] proposed a pattern technique by using active learning to import frequent patterns from external corpus. Li *et al.* [80] proposed a minimally supervised model for Chinese event extraction from multiple views, including *pattern similarity view* (PSV), *semantic relationship view* (SRV) and *morphological structure view* (MSV). Furthermore, each

[4]http://eventdata.parusanalytics.com/software.dir/tabari.html

[5]https://wwwnlpir.nist.gov/related_projects/muc/index.html

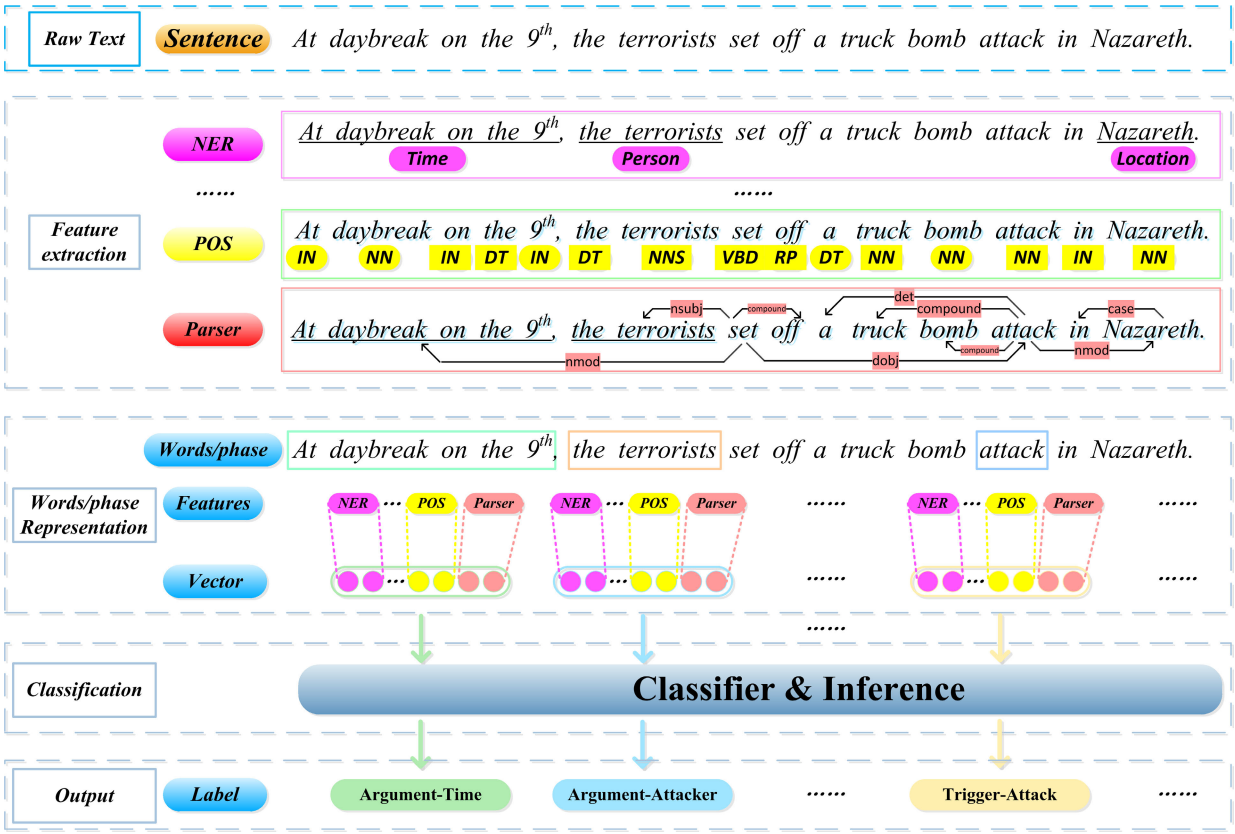


FIGURE 4. Illustration of event extraction based on machine learning. Word/phase features are obtained from execution feature engineering and then input to classifiers for event extraction to output trigger and arguments.

view can also be used to extract event patterns. The PSV ranks each candidate pattern according to its structural similarity to existing ones, and then accepts those top ranked news patterns. In contrast, the SRV captures the relevant event mentions from relevant documents while the MSV is incorporated to infer new patterns.

The pattern-based event extraction has been applied in many industrial applications for its high extraction accuracy from using high quality event patterns established by domain experts. However, constructing a large scale of event patterns is cost prohibitive. As such, recent years have witnessed a fast development of various machine learning-based event extraction techniques.

V. EVENT EXTRACTION BASED ON MACHINE LEARNING

This section reviews those using traditional machine learning algorithms, like *support vector machine* (SVM), *maximum entropy* (ME) and etc., for event extraction. Those using neural networks techniques (or so-called deep learning) will be reviewed in the next section. Note that they both are a kind of supervised learning techniques and require training data with ground truth labels, which is normally done by experts with professionals knowledge. How to annotate event and their arguments has been introduced in Section III.

The basic idea of machine learning approaches is almost the same, that is, learning classifiers from training data and applying classifiers for event extraction from new text. Fig. 4 illustrates the overall structure of event extraction based on machine learning. Furthermore, event extraction based on machine learning can be generally divided into two stages and four subtasks:

- Stage I includes two subtasks: **(1) trigger detection**, i.e., detect whether an event exists, and if so, the corresponding event trigger in a text; and **(2) trigger/event type identification**, i.e., classify the trigger/event as one of given event types;
- Stage II includes two subtasks: **(3) argument detection**, i.e., detect which entity, time, and values are arguments; and **(4) argument role identification**, i.e., classify the arguments' roles according to the identified event type.

The two stages and four subtasks can be executed in a *pipeline* manner, where multiple independent classifiers are each trained and the output of one classifier can also serve as a part of input to its successive classifier. The four subtasks can also be executed in a *joint* manner, where one classifier performs multi-task classification and directly outputs multiple subtasks' results.

For either pipeline or joint execution, learning classifiers need to first perform some *feature engineering* work, i.e., extracting features from texts as inputs to the classification model. So in this section, we first introduce some common features used for classifier training; Then we review

and compare the pipeline and joint classification approaches in the literature.

A. TEXT FEATURE FOR LEARNING MODELS

The mostly used text features can be generally divided into three types: *lexical*, *syntactic* and *semantic* feature. Most of them can be obtained via some open-source NLP tools.

Some commonly used lexical features include: (1) full word, lowercase word and proximity word; (2) lemmatized word, which returning different forms of a single word to its root form. For example, the word *computers* is an inflected form of *computer*; (3) POS tag, which marking up a word in a corpus as corresponding to a particular part of speech such as nouns, verbs, adjectives, adverbs and etc.

The syntactic features are obtained from dependency parsing, which is to work out the lexical dependency relation structure of sentences. In brief, it creates edges between words in the sentences denoting different types of relations and organizes them as a tree structure. Some commonly used syntactic features include: (1) the label of dependency path; (2) the dependency word and its lexical features; (3) the depth of candidate word in a dependency tree.

Some commonly used semantic features include: (1) synonyms in linguistic dictionaries and their lexical features; (2) event and entity type features, which are often used in argument identification.

Each feature can be represented as a binary vector based on the word-of-bag model. Feature engineering is about how to select the most important features and how to integrate them into a high dimensional vector to represent each word in a sentence. Finally, text features together with their corresponding labels from training datasets are used to train event extraction classifiers.

B. 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. To train classifiers, diverse features can be used, including local features within sentences and global features within documents. According to how local and global features are applied for classifier training, we divide pipeline classification approaches into sentence level event extraction and document level event extraction.

1) SENTENCE LEVEL EVENT EXTRACTION

In sentence level event extraction, a sentence is firstly tokenized into discrete tokens, and then each token is represented by a feature vector based on the result of feature engineering. In English, a token is normally a word; Yet in some languages, a token could be one character or multiple characters as a word or a phrase. Classifiers are trained from annotated text from a corpus and then used to determine whether a token is a trigger word (or event argument) and its event type (or argument role).

David Ahn [27] presented a typical pipeline processing framework consisting of two consecutive classifiers: The first one, called TiMBL, applies a *nearest neighbor* learning algorithm for detecting trigger(s); The second classifiers, called MegaM, adopts a *maximum entropy* learner for identifying argument(s). To train the TiBML classifier, lexical features, WordNet features, context features, dependency features and related entity features are used; and the features of trigger word, event type, entity mention, entity type, and dependency path between trigger word and entity are used for training the MegaM classifier. Many different pipeline classifiers have been proposed and they are trained by diverse types of features. For example, Chieu and Ng [85] added unigram, bigram and etc. as features and employed a maximum entropy classifier. In the biomedical domain, more domain-specific features and professional knowledge are exploited for classifier training, like frequency features, token features, path features and etc [86]–[93].

Some researchers have proposed to integrate the pattern matching into the machine learning framework [94]–[96]. As reviewed in the previous section, event patterns can provide more precise event structures with explicit relations between some particular triggers and their associated arguments, though such relations are kind of manually designed without too much extensibility. Grishman *et al.* [94], [97] proposed to first perform pattern matching so as to preassign some potential event types,[6] and then a classifier is applied to identify the remaining event mentions. The two approaches are complementary to each other to augment event extraction. Besides, pattern matching can also be executed after a machine learning classifier has detected a trigger [95]. Since event arguments are closely related to the trigger type, so applying some well-established trigger-argument relations can help improving the argument identification. Furthermore, event patterns can also be merged with token features as input for machine learning classifications [96].

As shown in Fig. 2, sentence tokenization is a trivial task in some languages, like English, as words are explicitly separated by delimiters. However, in some other languages, like Chinese [98]–[100] and Japanese [101], a sentence consists of consecutive characters without using delimiters to separate words. Therefore, *word segmentation* is normally required to firstly divide a sentence into many discrete tokens (words/phases). As the word segmentation can be implemented independent of the event extraction task, segmentation errors, if any, could be propagated to the downstream tasks and degrade their performance [99], [100]. Furthermore, due to the ambiguity of natural language, it is likely that after word segmentation, a word consists of two or more triggers; While a trigger is divided into two or more words [99], [100].

To solve such tokenization problems, Zhao *et al.* [98] proposed, for each word after sentence segmentation, to first obtain potential triggers each with its corresponding event

[6] https://nlp.cs.nyu.edu/meyers/GLARF.html

type from a dictionary of synonyms. Chen and Ji [99] proposed to employ both word-level and character-level trigger labeling strategy. Furthermore, they proposed to use character-level features, including the current character, its previous and next character and etc., to train a *maximum entropy Markov model* classifier for trigger detection. Li *et al.* [100] employed compositional semantics insider triggers and discourse consistency between trigger mentions to augment Chinese trigger identification. Specifically, a verb word is a trigger, if its components (one or more Chinese characters) are labeled as Chinese triggers. For discourse consistency, a single character of a verb word, if it belongs to labeled triggers, is merged with its previous and next character to form candidate triggers.

2) DOCUMENT LEVEL EVENT EXTRACTION

In the aforementioned approaches, only *local information* of the words or phrases within a sentence and sentence-level contexts are applied to train classifiers. However, if we put the event extraction task, even for extracting an event only from one sentence, against a larger background, like a document with multiple sentences or a collection with multiple documents, many *global information* are ready to be exploited to augment the extraction accuracy. For document level event extraction, two key design issues include: what kind of global information can be used; and how to apply them to assist event extraction. For the first design issue, global information, like word sense, entity type and argument mention, can be mined from cross-document, cross-sentence, cross-event and cross-entity inferences. For the second, global information can be used either as a complimentary module to local classifiers or as global features in local classifiers.

Global information can be exploited to build an additional inference model to augment local classifiers through evaluating the confidence of local classifiers' outputs [102]–[105]. Ji and Grishman [102] observed that event arguments would maintain some consistency across sentences and documents, like word sense consistency in different sentences and related documents, and consistency of arguments and roles across different mentions of the same or related events. To exploit such observations, they proposed to establish two global inference rules for a cluster of topically-related documents, namely, one trigger sense per cluster and one argument role per cluster, to help improving the extraction confidence of sentence-level classifiers. Liao and Grishman [103] later on extended such applications of global information inference and further proposed to apply document-level cross-event information. Liu *et al.* [105] explored two types of global information, namely, event-event association and topic-event association, to build a *probabilistic soft logic* (PSL) model for further processing the initial judgements from local classifiers to generate final extraction results.

Global information can also be exploited as global features together with sentence-level features to train local event extraction classifiers [106]–[108]. Hong *et al.* [106] proposed a cross-entity inference model to extract entity-type consistence as relation features. They argued that entities of the consistent type normally participate in similar events as the same role. To this end, they defined 9 new global features, including entity subtype, entity-subtype co-occurence in domain, entity-subtype of arguments and etc., to train a set of sentence-level SVM classifiers. Similar approaches have also been adopted in [107], [108]. Liao and Grishman [107] proposed to first compute topical distributions for each document via the *latent dirichlet allocation* (LDA) model [109] from a collection of documents and encoded such topic features into local extraction classifiers. In [108], three types of global features are defined, including lexical bridge features, discourse bridge features and role filler distribution features, and they are used together with local features to train extraction classifiers.

C. JOINT CLASSIFICATION MODEL

The aforementioned pipeline classification models could suffer from the *error propagation* problem, where errors in an upstream classifier are easily propagated to those downstream classifiers and could degrade their performance. On the other hand, a downstream classifier cannot impact on its previous classifiers' decisions, and inter-dependencies of different subtasks cannot be well exploited. To address such pipeline problems, *joint classification* have been proposed as a promising alternative to enjoy possible benefits from the close interactions between two or more subtasks, as useful information from one subtask can be carried both forward to next ones and backward to previous ones.

Joint classification models can be trained for multiple subtasks in each event extraction stage [110]–[112]. For example, Li *et al.* [110] proposed a joint model for trigger detection and event type identification, where an inference framework based on *integer logic programming* (ILP) is used to integrate two types of classifiers. In particular, they proposed two trigger detection models: One is based on the *conditional random field* (CRF) and another based on *maximum entropy* (ME). The ILP-based framework can help finding the optimal values for constrained variables to minimize a weighted objective function. Therefore, multiple subtasks can be jointly trained within the ILP inference framework yet each with different constraints and weights. Similar approaches have also been adopted by Li *et al.* [111] to train a joint model for discourse-level argument determination and role identification. Chen and Ng [112] trained two SVM-based joint classifiers each for one stage subtasks, yet with more linguistic features.

Joint classification models can also be trained to simultaneously extract the trigger and the corresponding arguments of an event according to predefined event structures [113], [114]. For example, Li *et al.* [113] formulate the event extraction as a structured learning problem, and proposed a joint extraction algorithm integrating both local and global features into a structured perceptron model [115] to predict event triggers and arguments simultaneously. In particular, the outcome of the entire sentence can be considered as a

graph in which trigger or argument is represented as node, and the argument role is represented as a typed edge from a trigger to its argument. They applied the beam-search to perform inexact decoding on the graph to capture the dependencies between triggers and arguments. Judea and Strube [114] observed that event extraction are structurally identical to the frame-semantic parsing which is to extract semantic predicate-argument structures from text. As such, they optimized and retrained the SEMAFOR [116],[7] a frame-semantic parsing system, for structural event extraction.

The structured prediction approaches have also been widely used in joint extraction models in biomedical domain [117]–[127]. For example, Riedel *et al.* [117], [118] represented events as relationally structured tokens of a sentence, and applied a joint probabilistic model based on *Markov logic* for biomedical event extraction. Venugopal *et al.* [119] combined the *Markov logic networks* (MLNs) and SVMs as a joint extraction model, where they leveraged SVM classifiers to handle high-dimensional features and modeled relational dependencies by MLNs. Vlachos *et al.* [122] employed a search-based structured prediction framework to provide high modeling flexibility. McClosky *et al.* [124] exploited the tree structures of event-argument in a re-ranking dependency parser to capture global event structure properties.

Besides event extraction, joint classification models can also be trained with other NLP tasks, like named entity recognition, event co-reference resolution, event relation extraction and etc. Some researchers have proposed to train a single model to jointly execute these tasks [128]–[132]. For example, Li *et al.* [128] proposed a framework to jointly execute these tasks together with event extraction in a single model. In particular, they used an information network to represent entities, relations, and events as an information network representation which extracts all of them by one single model based on structured prediction. The aforementioned joint extraction approaches only operate at sentence level but may miss valuable information from document level. On the basis of Li's model [128], Judea and Strube [129] presented an global inference model to incorporate the global and document context into the intra-sentential base system to extract entity mentions, events and relations jointly. In addition, Araki and Mitamura [131] found that events and their co-references offer useful semantic and discourse information for both tasks. They proposed a document-level joint model to capture the interactions between event trigger and event co-references so as to improve the performance of event trigger identification and event co-reference resolution simultaneously.

VI. EVENT EXTRACTION BASED ON DEEP LEARNING

Feature engineering is the main challenging issue of event extraction based on machine learning. As reviewed in the previous section, although diverse features like lexical, syntactic, semantic features and etc. can be crafted as classifiers' inputs, their construction requires linguistic knowledge and domain expertise, which might limit the applicability and adaptability of the trained classification models. Furthermore, 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.

Recently, deep learning techniques, which use multiple layers of connected artificial *neurons* to construct an artificial neural network, have been intensively studied for various classification tasks. In an artificial neural network, the lowest layer can take raw data with a very simple representation as its input. Each layer can learn to transform its lower layer input into a more abstract and composite representation, which is then input to its own higher layer, until the highest layer whose output feature is then used for classification. Compared with the classical machine learning techniques, deep learning can help to greatly reduce the difficulties of feature engineering.

Deep learning has been successfully applied in various NLP tasks, such as *named entity recognition* (NER) [133], search query retrieval and question answering [134], [135], sentence classification [136], [137], name tagging and semantic role labeling [138], relation extraction [139], [140]. For event extraction, many deep learning schemes have also been proposed recently [141]–[146]. 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).

How to design an efficient neural network architecture is the main challenging issue for event extraction based on deep learning. We will review some typical neural networks in the rest of this section. Before that, we briefly introduce the *word embedding* technique, as most neural networks for event extraction share a common approach of using word embedding as the raw data input. Word embedding techniques are used to convert a word or a phrase in the vocabulary into a low-dimensional and real-valued vector. In practice, the vector representation of a word is trained from a large scale corpus, and various word embedding models have been proposed, such as the *continuous bag-of words model* (CBOW) and *continuous skip-gram model* (SKIP-GRAM) [147], [148].

A. CONVOLUTIONAL NEURAL NETWORKS

One of the mostly used neural network structures is the *convolutional neural network* (CNN), which consists of multilayer fully connected neurons. That is, each neuron in a lower layer is connected to all neurons in its upper layer. As a CNN is capable of learning text hidden features based on the continuous and generalized word embeddings, it has been proven to be efficient to capture the syntactics and semantics of a sentence [136].

Nguyen and Grishman [141] might be the first researchers of designing a CNN for event detection, viz. identifying

[7]http://www.cs.cmu.edu/ ark/SEMAFOR/

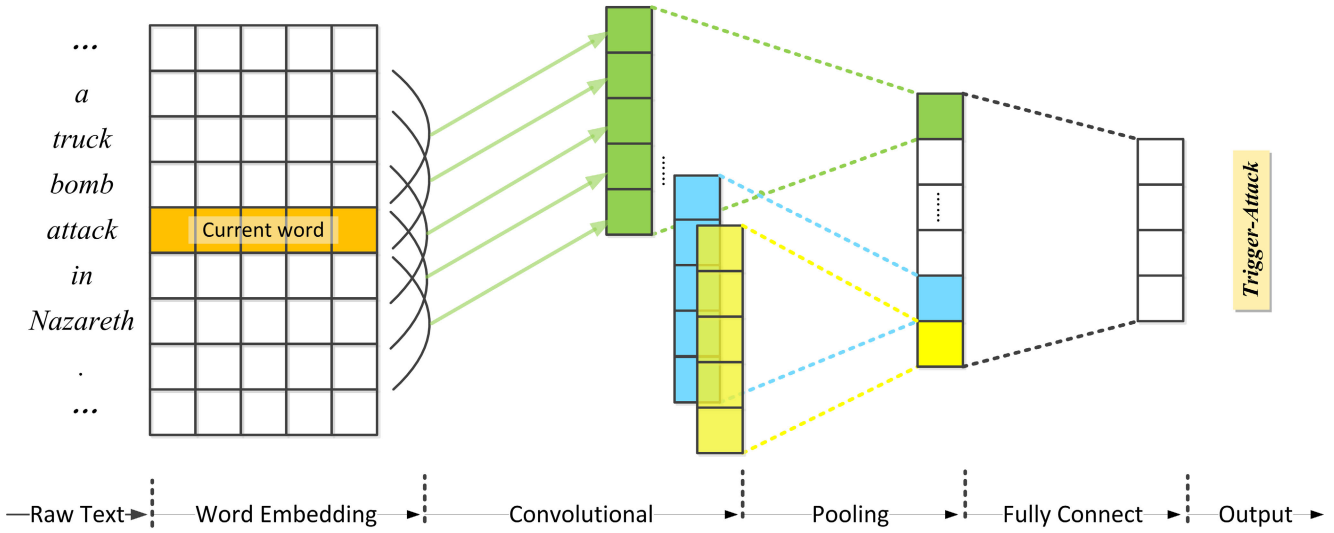


FIGURE 5. Illustration of using a convolution neural network for event extraction.

a trigger and its event type in a sentence. As illustrated by Fig. 5, each word is firstly transformed into a real-valued vector representation, which is a concatenation of the word embedding, its position embedding and entity type embedding, as the network input. The CNN consists of, from input to output, a convolution layer, a max pooling layer and a softmax layer, and outputs the classification result for each word. Many typical techniques can be used for training the CNN, including the back-propagation gradient, dropout regularization, stochastic gradient descent, shuffled mini-batches, AdaDelta learning rate adaptive and weight optimization.

The typical CNN structure often employs a max-pooling layer with a max operation over the representation of an entire sentence, however, a sentence may contain more than one event sharing arguments yet with diverse roles. Chen *et al.* [149] proposed a *Dynamic Multi-Pooling Convolutional Neural Network* (DMCNN) to evaluate each part of a sentence via a dynamic multi-pooling layer extracting both lexical-level and sentence-level features. In DMCNN, each feature map is divided into three parts according to the predicted trigger, and the max value of each part is kept to reserve more valuable information other than using a single max-pooling value. Furthermore, their CNN model also uses a skip-gram word model to capture meaningful semantic regularities for words.

In both [141] and [149], the convolutionary operation linearly maps the vectors for the k-grams into the feature space, where such k-gram vectors are obtained as the concatenation of the k consecutive words' representations. To also exploit long-range and non-consecutive dependencies, Nguyen and Grishman [150] proposed to perform convolutionary operations on all possible non-consecutive k-grams in a sentence, where the max-pooling function is used to compute convolution scores for distinguishing the most important non-consecutive k-gram for event detection.

Some other improvements of the typical CNN model have also been proposed [151]–[154]. For example, Burel *et al.* [152] designed a semantically enhanced deep-learning model, called Dual-CNN, which adds a semantic layer in a typical CNN to capture contextual information. Li *et al.* [153] proposed a *parallel multi-pooling convolutional neural network* (PMCNN), which can capture the compositional semantic features of sentences for biomedical event extraction. The PMCNN also utilizes dependency-based embedding for word semantic and syntactic representations and employs a rectified linear unit as a nonlinear function. Kodelja *et al.* [154] built the representation of global contexts following a bootstrapping approach, and integrated the representation into a CNN model for event extraction.

B. RECURRENT NEURAL NETWORKS

Many CNN-based event extraction schemes suffer from the error propagation problem due to their pipeline execution of the two subtasks, namely, event detection first and argument identification second. Furthermore, the CNN structure normally takes the concatenation of words' embedding as input, and the convolutionary operation is executed for consecutive words to capture contextual relations of the current word to its neighboring words. As such, they cannot well capture some potential interdependencies in between distant words to exploit a sentence as a whole for jointly extracting trigger and arguments; While joint extraction, as discussed in previous section, can exploit the relations between event trigger and event arguments to reciprocate the two individual subtasks.

In language modelling, a sentence is often regarded as a sequence of words, i.e., one word after another from the start to the end of a sentence. The *recurrent neural network* (RNN) structure, which consists of a series of connected neurons, can effectively make use of such sequential inputs. As illustrated by Fig. 6, a simple RNN consists of a series of connected *long and short term memory* (LSTM) neurons, where the output of a LSTM neuron also serves as the input of its sequentially connected LSTM neuron. As such, the RNN structure

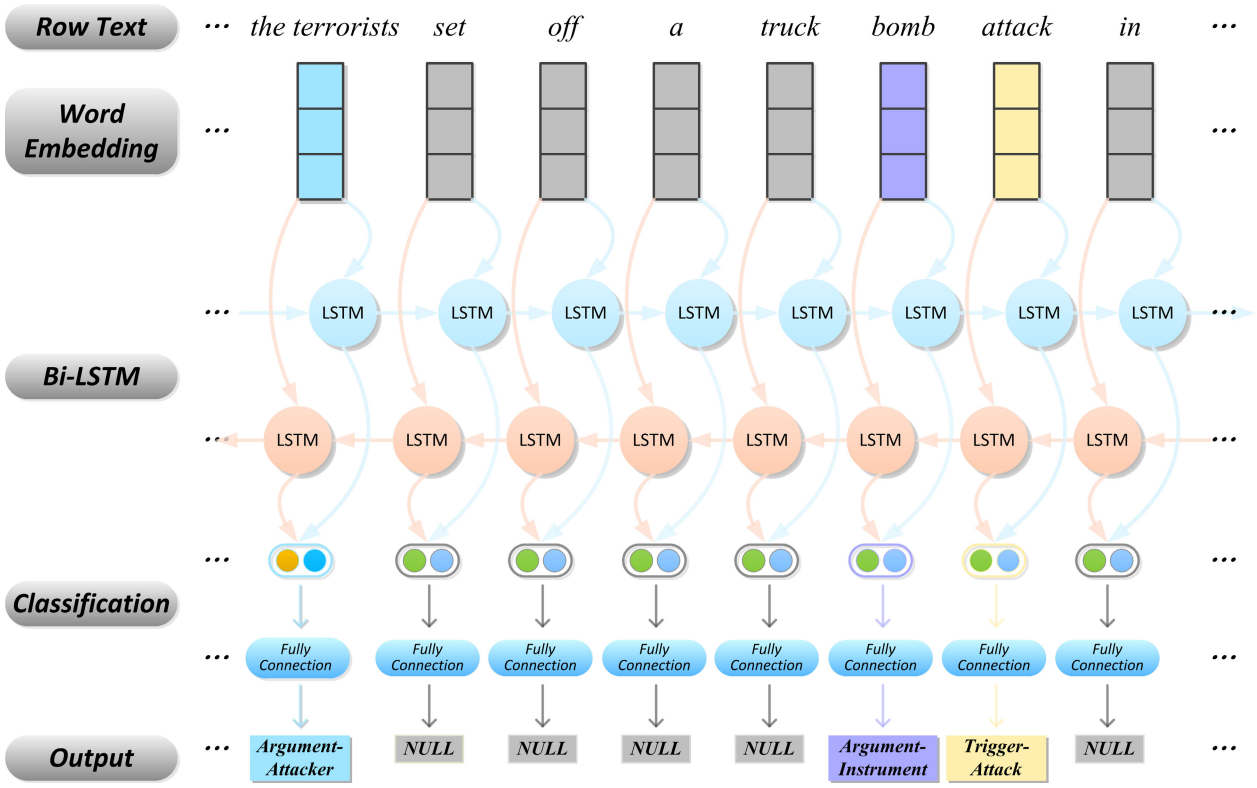


FIGURE 6. Illustration of using a recurrent neural network for event extraction.

can exploit the potential dependencies in between any two words, either directly or indirectly connected, which enables its wide applications in many NLP tasks [155], including named entity recognition [156], part-of-speech tagging [157], relation extraction [158], sentence parsing [159], sequence labeling [160] and etc. Furthermore, the 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.

For event extraction, some RNN-based models have been proposed to exploit words' interdependencies by inputting words according to their sequential order, either forwardly or backwardly, in a sentence [161]–[164]. For example, Nguyen *et al.* [161] designed a bi-directional RNN architecture for joint event extraction, as illustrated by Fig. 6. Their model runs over sentences in both forward and reverse direction by two individual RNN, each consisting of a series of *gated recurrent units* (GRU) [165]. The joint extraction consists of two phases: the encoding phase and the prediction phase. In the encoding phase, different from the CNN model, it does not employ the position features but replacing them with binary vectors to represent the dependency features for predict event triggers and arguments jointly. In the prediction phase, the model classifies the dependencies between trigger and argument into three categories: (1) the dependencies among trigger subtypes; (2) the dependencies among argument roles; (3) the dependencies between trigger subtypes and argument roles.

Syntactic dependency in between words can also be used to augment the basic RNN structure. For example, Sha *et al.* [166] designed a dbRNN (*dependency bridge* RNN) by adding a syntactic dependent connection of two RNN neurons into a bidirectional RNN. As shown in Fig. 7, the syntactic dependency of Sentence 1 is exploited by including dependency bridges into the bidirectional RNN.

Besides using dependency bridges, the syntactic dependency tree of a sentence can also be directly exploited to build a tree-structured RNN [167]. Upon the typical Bi-LSTM (*bidirectional* LSTM), Zhang *et al.* [168] further constructed a Tree-LSTM yet centered at the target word by transforming the original dependency tree of a syntactic dependency analyzer for Chinese event detection. Li *et al.* [169] proposed to further augment a Tree-LSTM with external entity ontological knowledge for biomedical event extraction.

The RNN structure can be applied not only to sentence level but also document level event extraction. Duan *et al.* designed a DLRNN model (*document level* RNN) [170] to extract cross-sentence or even cross-document clues by using a distributed vector for document representation, which is to capture the topic distribution of a document by the unsupervised learning PV-DM model [171]. All the words in a document use the same document vector, and the concatenation of word embedding and document vector is used as the input of a Bi-LSTM model.

The RNN structure can also be applied to train a joint classification model for not only event extraction including trigger detection and argument identification, but also entity

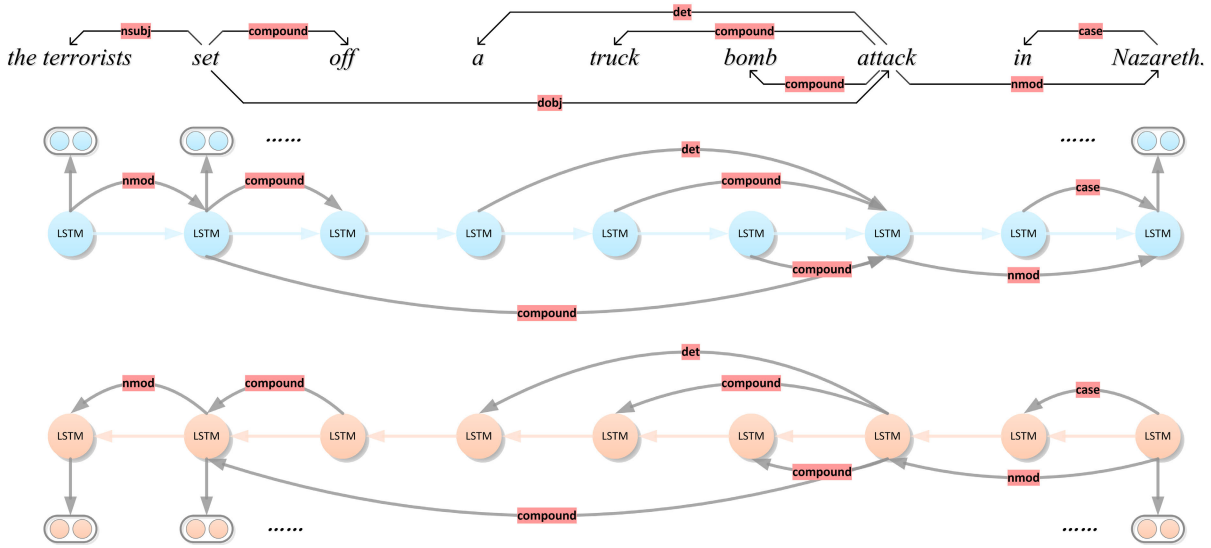


FIGURE 7. The dependency bridge on LSTM. Apart from the last LSTM cell, each cell also receives information from former syntactically related cells.

mention detection [172]. The three subtasks are executed in a pipeline way, in the order of entity mention detector, trigger classifier and argument role classifier, by training a single Bi-GRU (*bidirectional* GRU) network model, where the network hidden representations are shared for all the three subtasks to exploit some common knowledge across subtasks and potential dependencies or interactions in between the subtasks.

The aforementioned RNN structures have adopted the GRU or LSTM as the basic composing unit, which uses a gating strategy to control the information processing in the neural network. However, the gate computation cannot be done in parallel and is time-consuming. Zhang *et al.* [173] proposed to use a kind of *simple recurrent unit* (SRU) as the basic composing neuron for its capable of reducing gate computation complexity without incurring the multiplication operation dependent on previous units [174]. They built two bidirectional SRU models (Bi-SRU): one for learning word-level representations, and another for character-level representations.

C. GRAPH NERUAL NETWORKS

Recently, many neural networks operating on graphs, or called *graph neural networks* (GNNs) as a general reference, have been recently prompted for a wide range of application domains [175]–[177]. Simply put, a GNN applies multiple neurons operating on a graph structure to enable so called geometric deep learning in non-Euclidean spaces. Traditional neuron operations, like recurrent kernels and convolutional kernels that have been widely used in RNNs and CNNs, can also be applied in a graph structure so as to learn various deep features embedded within a graph for diverse tasks.

Some researchers have attempted to apply GNN models for event extraction [178]–[180]. The core issue of such approaches is to first construct a graph for words in text. Rao *et al.* [178] adopted a semantic analysis technique, called *abstract meaning representation* (AMR) [181], 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. Furthermore, Rao *et al.* argued that an event structure is a subgraph of an AMR graph and casted the event extraction task as a subgraph identification problem. They trained a graph LSTM model to identify such an event subgraph for biomedical event extraction.

Another approach of graph construction is based on some transformation for the syntactic dependency tree of a sentence [179], [180]. Notice that the syntactic dependency can be regarded as a directed edge from a head word to its dependent word, yet also with an edge label as the dependency type. At first, for each word, a new self-loop edge as proposed in [182] is created and included into the dependency tree, that is, an edge starting from a word and ending at the word. All such self-loop edges are with a same new edge type. Then for each dependency edge (w_i, w_j) with edge type T from word w_i to word w_j, a reverse edge (w_j, w_i) with a type T' is created and included into the dependency tree also. The graph construction is illustrated in Fig. 8. Based on the constructed graph, multi-hop graph convolutionary operations are conducted to capture the contextual characteristics from not only local neighboring words but also long-range dependent words for generating a new representation vector for each word.

D. HYBRID NEURAL NETWORK MODELS

The aforementioned three basic types of neural network architecture each has its own merits and demerits when used to capture diverse features, relations and dependencies in text for event extraction. Many researchers have proposed hybrid neural network models which combine different neural networks to enjoy each excellence. A common approach

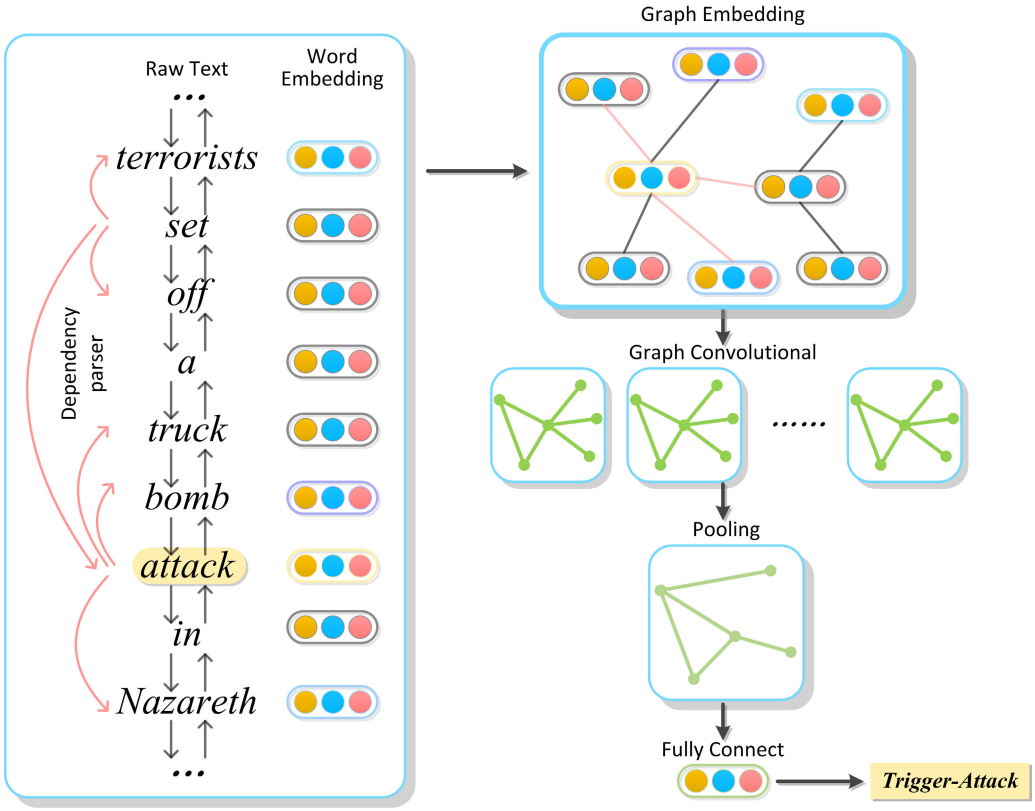


FIGURE 8. Illustration of using a graph convolution neural network for event extraction.

of building such a hybrid model is to use different neural networks to learn different types of word representations. For example, both [179] and [180] have firstly applied a Bi-LSTM model to obtain initial word presentations before performing graph convolutionary operation.

Some have proposed to combine the CNN and RNN as a hybrid model [183]–[189]. For example, Zeng *et al.* [183] proposed to first use a CNN for learning the local contextual representation of each word, which is concatenated with the output of another Bi-LSTM to obtain the final word representation for classification. Similar approaches of concatenating CNN output and Bi-LSTM hidden layers have also been applied to event extraction for Chinese, India and other languages [184]–[187]. Nurdin and Manlidevi [188] also designed a hybrid model consisting of a CNN and a Bi-LSTM for extracting events from Indonesian news with the following arguments: *who, did what, when, where, why* and *how*. Liu *et al.* [189] also used a CNN to obtain the local contextual representation for each word; Yet they proposed to use a Bi-LSTM to obtain a document representation as a weighted sum of concatenated hidden states of the forward and backward layers. The word local representation and document representation are then further concatenated for event trigger detection.

Another important type of hybrid model is the *generative adversarial network* (GAN) [190], which normally consists of two neural networks contesting each other, one dubbed as a *generator* G_{nn} and another a *discriminator* D_{nn}. The G_{nn} produces candidate results; Yet the D_{nn} evaluates them from the training data ground truth. The training process for G_{nn} introduces some noisy data fabricated from the training dataset; While the D_{nn} struggles to minimize the difference to the true data distribution. The generator and discriminator normally apply a same neural network structure, yet many GAN models have proposed different training strategies.

Some researchers have proposed to apply the GAN framework for event extraction [191]–[193]. They adopted the RNN structure like Bi-LSTM for the generator and discriminator network. In the training process, Hong *et al.* [191] proposed to regulate the learning process with a two-channel self-regulated learning strategy. In the self-regulation process, the generator is trained to produce the most spurious features; While the discriminator with a memory suppressor is trained to eliminate the fakes. Liu *et al.* [192] proposed an adversarial imitation strategy to incorporate a knowledge distillation module into the feature encoding procedure. In their hybrid model, a *teacher encoder* and a *student encoder*, each being a Bi-GRU network, are used. The teacher encoder is trained by gold annotations, yet the student encoder is trained by minimizing the distance between its output to that of the teacher encoder via an adversarial imitation learning. Zhang *et al.* [193] used the *reinforcement learning* (RL)

strategy to update a Q-table during the training process, where the Q-table records the reward values computed from the system states and actions.

E. ATTENTION MECHANISM

Attention mechanism first appeared in the field of computer vision, and its purpose was to emulate the visual attention mechanism of human brain. Recently, the attention mechanism has been widely used in many NLP tasks [194]–[196]. Simply put, attention is a discrimination mechanism to guide a neural model to unequally treat each component of the input according to its importance to a given task. Although it is implemented by assigning different weights to different neurons' states or outputs, the weights are actually self-learned from the model training process.

Many word-level attention mechanisms have been proposed to learn the importance of each word in a sentence, like distinguishing different argument words, word types, word relations [197]–[202]. They mainly differ in what elements should be given more attentions and how to train attention vectors. For example, Liu *et al.* [197] argued that the argument words to a trigger should receive more attentions 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. Wu *et al.* [198] applied argument information to train attentions with a Bi-LSTM network.

In word-level attention mechanisms, entity relations from syntactic parser can also be used to train attentions [199], [200]. The basic idea is that the syntactic dependency can provide the connection in between two possibly nonconsecutive yet distant words; While the dependency type can help to distinguish the syntactic importance in between words. For Chinese event extraction, as there has no explicit word segmentation like English, Wu *et al.* [202] also proposed a character-level attention mechanism to distinguish each character importance in a Chinese word.

Some researchers have also proposed to integrate both word-level and sentence-level attentions to augment event extraction in multi-sentence documents [203]–[205]. Zhao *et al.* [203] argued that in many multi-sentence documents, sentences in one document are often correlated with respect to the document theme, although they may contain different types of events. They proposed a DEEB-RNN model with a hierarchical and supervised attention mechanism, which pays word-level attention to event triggers and sentence-level attention to those sentences containing events. To this end, they constructed two gold attention vectors: one for word-level attention based on the sentence trigger; another for sentence-level attention for each sentence if containing a trigger word. Besides sentence interdependencies in one document, it is could also be the case that multiple events are embedded within a single sentence. Chen *et al.* [204] argued that events mentioned in the same sentence tend to be semantically coherent. To capture both intra-sentence dependency and inter-sentence correlation, they proposed a HBTNGMA model with a gated multi-level attention mechanism for extracting and fusing intra-sentence and inter-sentence contextual information to augment event detection.

Besides exploiting words and sentences, some researchers have proposed to integrate extra knowledge for attentions, like using multi-lingual knowledge [206] or a priori trigger corpus [207]. Most event extraction models are trained and applied for a particular language, which may suffer from the typical ambiguity problem of a single word with different meanings in different contexts. Liu *et al.* [206] examined the annotated events in ACE 2005 and observed that 57% of the trigger words are ambiguous. They argued that using a multilingual approach can help to deal with the ambiguity problem based on the observations of *multilingual consistency* and *multilingual complementation*. As such, they proposed a *gated multilingual attention* frame which contains both mono-lingual context attention and gated cross-lingual attention. For the latter, they first applied a machine translator to obtain the translated text in another language. Li *et al.* [207] designed a prior knowledge integration network to encode their collected keywords as a prior knowledge representation. Such knowledge representations are then integrated with a self-attention network structure.

VII. EVENT EXTRACTION BASED ON SEMI-SUPERVISED LEARNING

The aforementioned algorithms based on machine learning and deep learning techniques are kind of supervised learning approaches, which require a labeled corpus for model training. As deep learning approaches normally involve a large number of parameters in a neural network, generally the larger the labeled corpus, the better a model can be trained. However, obtaining labeled corpus is a rather cost prohibitive task for its time-consuming and labor-intensive annotation process, which in most cases also requires domain expertise and professional knowledge. Due to this, many labeled corpus are with small size and low coverage. For example, in the ACE 2005 corpus, only 33 event types of interested were defined, a very low coverage for diverse applications; Furthermore, among the labeled events, about 60% event types have less than 100 instances, and 3 event types have even fewer than 10 instances.

How to improve the extraction accuracy from a small set of labeled gold data has become a critical challenge. A straightforward solution is to first automatically produce more training data, and then use mixture data containing both original gold data and newly generated ones for model training. In this section, we review such solutions in the literature, though they may have used different names, like semi-supervision, weak supervision, distant supervision and etc. We note that

although using mixture data impacts on the model training process, the basics of these learning-to-classify algorithms are similar to those reviewed in the previous two sections. In this section, we mainly focus on how they expand a small set of labeled data to a larger corpus, and how extraction models can be trained from mixture data.

A. JOINT DATA EXPANSION AND MODEL TRAINING

The set of labeled gold data may be small, however, they can be iteratively used for model training. Such an iterative model training via data replacement is a variant of the kind of well-known technique, called *bootstrapping* [208]. The basic idea is to first train a classifier with a small set of labeled data to classify new unlabeled data. Besides classified labels, a classifier also outputs classification confidence for new data. Then the new data with very high confidence can be included to the training data for next round model training.

The key challenge of employing classification results for data expansion lies in how to evaluate classification confidence for new data [209]–[212]. As events have complicated structures, like different event types containing different arguments with different roles, computing event extraction confidence is normally with low accuracies. Liao and Grishman [209], [210] proposed to use only a part of extraction results from new samples for data expansion, in particular, the most confident trigger with its most confident argument. Based on their extraction model [94], the most confident ''<role, trigger>'' pairs are selected based on the product between probability from their trigger classifier and argument classifier. Moreover, they employed an information retrieval system, called INDRI [213], to collect a cluster of related documents, and applied the cross-document inference algorithm proposed in [102] to include new training data with high confidence.

Wang *et al.* [211] proposed to use trigger-based *latent* instances to discover unlabeled data for expansion. If a word serves as a trigger in the gold dataset, all instances mentioning this word in the unlabeled dataset might also express the same latent instance. Based on such assumptions, they proposed an adversarial training process to filter out noisy instances yet distilling informative instances to include new data. Although including new data from classifiers' outputs is cost-effective, it actually cannot guarantee that the newly included data are with completely correct annotations. In this regard, manual checking is of great necessities. Liao and Grishman [212] proposed an *active learning* strategy, called pseudo co-testing, to employ minimum manual manpower to only annotate those high confidence new data.

The aforementioned algorithms have focused on the expansion of training data, yet with the same event type as the labeled gold ones. Recently, some researchers have proposed a kind of *transfer learning* algorithms to expand training data with event types different from the reference data in the gold dataset [214]–[218]. For example, Nguyen *et al.* [216] proposed a two-stage algorithm to train a CNN model that can effectively transfer knowledge from old event types to the target type (new type). Specifically, the first stage is to train a CNN model based on labeled data of old event types with randomly initialized weight matrices. The second stage is to train the CNN model based on a small set of labeled data of target types with the weight matrices initialized by the first stage. Finally, the two-stage trained CNN model is used for both target type and old type event detection.

Besides utilizing a small set of labeled data with new types, Huang *et al.* [218] proposed a zero-short transfer learning for extracting events with new unseen types, which only needs a manually structured definition of new event types (e.g., event type names and argument role names from event schema). In particular, they first constructed the vector representation of *event mention structure* (trigger, arguments and their relation structure from event instance) and *event type structure* (type, roles and their relation structure from event schema) via training a CNN model based on the gold data. They next use the optimized CNN model to represent event mention structure and event type structure of new data, and find the closest event type for each new event mention.

B. DATA EXPANSION FROM KNOWLEDGE BASES

Many existing knowledge bases store a large amount of structured information, such as FrameNet,[8] Freebase,[9] Wikipedia[10] WordNet,[11] which can be exploited to generate new labeled data as training data for event extraction.

The FrameNet defines many complete semantic frames, each of which consists of a lexical unit and a set of name elements. Such frames share highly similar structures with events. As many frames in FrameNet actually express certain events, some researchers have explored mapping frames to events for data expansion [219]–[221]. For example, Liu *et al.* expanded [220] the ACE training data by using events detected from given exemplar sentences in FrameNet. Specifically, they first learned an event detection model based on the ACE labeled data, which is then used to yield initial judgements for exemplar sentences. Then, a set of soft constraints are applied for global inference based on such hypotheses: ''*The same lexical unit, the same frame and the related frames tend to express the same event*.'' The initial judgements and soft constraints are then formalized as first-order formulas and modeled by *probabilistic soft logic* (PSL) [222] for global inference. Finally, they detect events from given exemplar sentences for data expansion.

Analogously, the *compound value types* (CVTs) in the Freebase (a semantic knowledge base) can be regarded as event templates, and CVTs instances are regarded as event instances. The types, values and roles of CVTs are regarded as event types, arguments in events and roles of arguments playing in events, respectively. Zeng et al [223] exploited structural information of CVTs in Freebase to automatically

[8] https://framenet.icsi.berkeley.edu/fndrupal/
[9] https://developers.google.com/freebase/
[10] https://www.wikipedia.org
[11] https://www.wordnet.org

annotate event mentions for data expansion. They first identified the key arguments from CVTs, which play an important role in one event. If a sentence contains all key arguments of a CVTs, it is likely to express the event presented by the CVTs. As a result, they recorded the words or phrases in this sentence to match the CVTs properties as the involved arguments with their roles for annotation.

Moreover, Chen *et al.* [221] proposed to expand training data by exploiting both FrameNet and Freebase. Araki and Mitamura [224] utilized the WordNet and Wikipedia to generate new training data. In addition to using the general knowledge bases, some studies have focused on using relevant knowledge bases for domain-specific event extraction [29], [178], [225], [226]. For example, Rao *et al.* [178] used the *biological pathway exchange* (BioPax) knowledge database, which contains relations between proteins, to expand training data from PubMed[12] central articles. In the financial domain, Yang *et al.* [29] utilized a financial event knowledge database for data expansion, which contains 9 common financial event types.

C. DATA EXPANSION FROM MULTI-LANGUAGE DATA

Motivated by the facts that a same event may be described in different languages and the labeled data from one language is highly possible to convey similar information in another language, some approaches have been proposed to utilize such multi-language information to address the data sparseness and low-coverage problem [227]–[231]. For example, Zhu *et al.* [227] employed Google Translate to eliminate the language gap between Chinese and English, and produced uniform text representations with bilingual word features. In this manner, it can merge the training data from both languages for model training.

In comparison, some researches have proposed to bootstrap event extraction via exploiting cross-lingual data, i.e. cross-lingual bootstrapping, without using machine translation or manually aligned knowledge base [228]–[230]. For example, Chen and Ji [228] proposed a co-training bootstrapping framework, which contains two monolingual self-training bootstrapping event extraction systems, one for English and another for Chinese. A labeled event can be transformed from one language to another, so as to produce so called projected triggers and arguments for event extraction in another language. Hsi *et al.* [229], [230] have proposed to augment a standard event extraction pipeline of classifiers by leveraging multilingual training via the use of multilingual features, such as universal POS tags, universal dependencies, bilingual dictionaries, multilingual word embeddings and etc.

VIII. EVENT EXTRACTION BASED ON UNSUPERVISED LEARNING

Unlike supervised and semi-supervised learning, unsupervised learning does not train event extraction models based on labeled corpus. Instead, unsupervised learning approaches mainly focus on open-domain event extraction tasks, like detecting trigger and arguments based on word distributional representations and clustering event instances and mentions according to their similarities.

A. EVENT MENTION DETECTION AND TRACKING

Event mention is collective of *keywords* that can describe an event from one or more sentences. The tasks include detecting event mentions in an article and tracking similar event mentions in different articles. Notice that classifications of trigger type and/or argument role are not required, as types and roles are usually not predefined in such tasks.

The TDT program [15] defined a *topic* as a set of news and/or stories that are strongly related by some seminal real-world event. It then defined the TDT task as determining whether a given article is related to a clustering of events of the same topic and provided a TDT corpus with simple labels like {Yes, No, Brief } for each article to indicate its relevance to a topic, where Brief indicates partially relavent.

In many TDT algorithms, sentences are firstly converted into vector representations, and the vector distance is computed to measure the similarity to some topic for event detection [232]–[236]. For example, Yang *et al.* [232] and Nallapati *et al.* [233] represented documents by their TF-IDF vectors, a conventional vector space model which uses the bag-of-terms representation. Specifically, terms (words or phrases) in documents are statistically weighted using the *term frequency* (TF) and *inverse document frequency* (IDF). They keep the k top-ranking terms in bag-of-terms as a representation vector for a document.

Stokes and Carthy proposed to represent documents by the *lexical chains* [237], which explore the *cohesion structure* of text to create a sequence of semantically related words [234]. For example, the lexical chains of a document concerning *airplane* might consist of the following words: *plane, airplane, pilot, cockpit, airhostess, wing, engine*. They identified the lexical chains in documents using WordNet, which represent synonymous words in terms of a single unique identifier.

Following the task definition of the TDT program, some other researches have been conducted to detect whether new articles in various websites are related to some already identified event, without using the TDT corpus [32]–[37], [238]. For example, Naughton *et al.* [35] proposed to vectorize sentences using the bag-of-words encoding from news articles and clustered sentences via using the agglomerative hierarchical clustering algorithm [239].

Besides using word and sentence embeddings, some have proposed to exploit several additional information of news articles, like time and location, to augment event mention detection. Ribeiro *et al.* [32] proposed to integrate time, location and content dimensions into text representation, and applied an *all pairs similarity search* algorithm and a Markov clustering algorithm to cluster news articles describing the same event. Likewise, Yu and Wu [33] proposed to a Time2Vec representation technique that constructs article

[12]https://www.ncbi.nlm.nih.gov/pubmed

representation by a context vector and time vector and employed a dual-level clustering algorithm for event detection.

Recently, several novel methods have been proposed for event detection and tracking, including the negative-examples-pruning support vector machine [238], multiple instance learning model based on convolutional neural networks [36], and weighted undirected bipartite graph based ranking framework [34].

B. EVENT EXTRACTION AND CLUSTERING

The task of event mention detection mainly focuses on detecting event keywords in order to cluster sentences or articles expressing a same event. Some studies have proposed to further discriminate event trigger from arguments for clustering similar events and constructing event schema for similar events [44]–[51].

A straightforward approach is to regard the verb of a sentence as an event trigger. For example, Rusu *et al.* [47] 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. Moreover, some knowledge bases can be applied to augment trigger and argument discriminations. Chambers and Jurafsky [44] considered verbs and their synset in WordNet as triggers; While the entities of syntactic objects are arguments. Huang *et al.* [45] considered all noun and verb concepts in OntoNotes[13] and verbal and lexical units in FrameNet as candidate event triggers. Furthermore, they regarded all concepts with semantic relations to candidate triggers as candidate arguments. Then they computed the similarity between each pair of candidate trigger and arguments for identifying final trigger and arguments.

Based on the extracted triggers and arguments, event instances can be clustered into different event groups each with a latent yet distinguished topic. For example, Chambers and Jurafsky [44] proposed to use a *probabilistic latent Dirichlet allocation* topic model to compute event similarities for their clustering. Romadhony *et al.* [49] proposed to utilize structured knowledge bases to cluster triggers and arguments. Jans *et al.* [240] identified and clustered *event chains*, which can be viewed as a partial event structure consisting of a verb and its dependency actor based on the skip-gram statistics.

Furthermore, for each event group, an *event schema* can also be established with a *slot-value structure*, where a *slot* can represent some argument role and a *value* the corresponding argument of an event instance. Yuan *et al.* [46] first introduced a new event profiling problem to fill a slot-value event structure from open-domain news documents. They also proposed a schema induction framework by exploiting entity co-occurrence information to extract slot patterns. Glavasŝ and Ŝnajder [52] argued that events often contain some common argument roles like agent, target, time and location and proposed to construct an event graph structure to identify a same event yet mentioned in different documents.

[13]https://catalog.ldc.upenn.edu/LDC2013T19

C. EVENT EXTRACTION FROM SOCIAL MEDIA

Many online social networks, such as Twitter, Facebook and etc., present a large number of most up-to-date information. Twitter is a representative one. It has been reported that there are about 200 million tweets posted every day [241] and 1% of tweets covers 95% of events also reported on newswire [242]. We next take Tweet as an example and review some work on event extraction from social media.

Compared with newswire articles, Tweet posts have its own characteristics and event extraction faces new challenges. Tweets are mostly published by individual users, and each tweet is with a character limit. So tweets are often with abbreviations, misspellings and grammar errors, which causes many fragmented and noisy text without enough contexts for event extraction [28], [38]–[41]. In contrast to event extraction from newswire, entities, date, location, and keywords are the main components to be extracted from tweets. Since the tweets are short, all entities in tweets are considered as participants of an event of interest.

Weng and Lee [42] proposed to first analyze individual words with their frequencies and filtered out trivial words based on their correlations. After that, remaining words are clustered to form events by a modularity-based graph partitioning technique. Ritter *et al.* [38] utilized a named entity tagger and the TempEx tool [243] to resolve the temporal expressions. Zhou *et al.* [40], [41] proposed to filter out noisy tweets through lexicon matching. The lexicon contains event keywords (or triggers) which are extracted from newswire articles published about the same period as tweets. Furthermore, Zhou *et al.* [40] proposed to first identify named entities from newswire and built a dictionary to match the named entities in tweets. In particular, they employed the SUTime [244] to resolve the ambiguity of time expressions and proposed an unsupervised *latent event model* to extract events from tweets. Guille *et al.* [43] proposed a mention-anomaly-based event detection algorithm by computing the occurrence anomaly in the frequency of a word for a given contiguous sequence of time-slice to detect event.

IX. EVENT EXTRACTION PERFORMANCE COMPARISON

As reviewed in previous sections, event extraction may have different task definitions and may have been experimented on different corpora, As such, a fair comparison is not likely to be conducted for all algorithms. But thanks to those public evaluation programs, the standardization of performance metrics and open datasets have made it possible for researchers to compare their algorithms. In this section, we mainly compare the algorithms experimented on the public ACE 2005 dataset with the standard evaluation procedures as follows:

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%	-	-
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%	-	-

- **Trigger detection:** A trigger is correctly detected if its offsets (viz., the position of the trigger word in text) match a reference trigger.
- **Type identification:** An event type is correctly identified if both the trigger's offset and event type match a reference trigger and its event type.

TABLE 5. Performance comparison of Chinese event extraction on the ACE 2005 dataset.

System	Approach	Results-F1			
		Trigger	Event Type	Argument	Argument role
Zhao et al. (2008) [98]	Machine Learning	-	61.2%	-	64.6%
Chen & Ji (2009) [99]	Machine Learning	62.7%	59.9%	46.5%	43.8%
Chen et al. (2012) [112]	Machine Learning	66.7%	63.2%	49.5%	44.6%
Li et al. (2012) [100]	Machine Learning	70.5%	66.9%	55.3%	50.8%
Li et al. (2012) [110]	Machine Learning	73.1%	70.2%	58.1%	53.9%
Li et al. (2013) [111]	Machine Learning	-	-	63.2%	57.4%
Li et al. (2013) [104]	Machine Learning	-	-	64.4%	58.7%
Li et al. (2016) [80]	Pattern Matching	-	58.4%	-	-
Ghaeini et al. (2016) [162]	Convolutional Neural Networks	69.0%	64.8%	-	-
Zeng et al. (2016) [184]	Hybrid Neural Networks	69.3%	64.5%	52.6%	46.9%
Feng et al. (2016) [185]	Hybrid Neural Networks	68.2%	63.0%	-	-
Zhang et al. (2018) [168]	Recurrent Neural Networks	70.9%	63.9%	-	-

- **Argument detection:** An argument is correctly detected if its offsets match any of the reference argument mentions (viz., correctly recognizing participants in an event).
- **Role identification:** An argument role is correctly identified if its event type, offsets, and role match any of the reference argument mentions.

Table 4 and Table 5 present the reported event extraction results by different algorithms in English and Chinese dataset, respectively, in terms of standard F-Measure (F1) which is obtained by the two commonly used performance metrics, viz., Precision and Recall. We note that not all algorithms reported all the results, as some of them were only designed for implementing some subtasks. It is also worth of noting that event extraction also depends on some upstream tasks' results, like named entity recognition entity mention classification and etc. Most of these algorithms have assumed to directly use the gold annotations for entities, times, values and etc. in ACE 2005 as a part of the input to the event extraction task. However, we note that the ACE 2005 is a small dataset even with a very few of erroneous annotations.

From the table, we can observe that in general, algorithms on English can achieve better performance than those on Chinese. This may be due to that event extraction for Chinese sentences is also dependent on the word segmentation results. Furthermore, it can also note that recent advanced neural models can achieve better performance, which can be attributed to their powerful capabilities for learning deep yet more comprehensive context-aware and/or syntactic-aware word and sentence representations. Finally, we can observe that all subtasks have not achieved very high F1 values, which, on the one hand, indicates the difficulties of event extraction tasks, and on the other hand, motivates more advanced algorithms to be developed.

X. CONCLUSION AND DISCUSSION

Event extraction is an important task in natural language processing, with the objectives of detecting whether sentences have mentioned some real-world event, and if so, classifying event types and identifying event arguments. For its diverse applications, event extraction has been intensively researched decades ago, and recently it has been attracting more than ever research interests due to the fast development of many novel techniques like deep learning.

In this article, we have tried to provide a comprehensive yet up-to-date review for event extraction from text. We first introduced the public evaluation programs as well as their task definitions and annotated datasets for both closed-domain and open-domain event extraction. We divided the solution approaches into five groups, including pattern matching algorithms, machine learning methods, deep learning models, semi-supervised learning techniques, and unsupervised learning schemes. We have presented and analyzed the most representative methods in each group, especially their origins, basics, strengths and weaknesses. In addition, we introduced evaluation methods and compared typical algorithms experimented on the ACE 2005 corpus.

The pattern matching approaches normally can achieve high extraction accuracy, however, the pattern construction is with the prohibitive cost of human efforts and professional knowledge. The supervised learning approaches including machine learning and deep learning, on the other hand, are based on the availabilities of large annotated corpus, although they seems to achieve better performance. Furthermore, these closed-domain extraction algorithms are normally trained for specific domains, and may not be directly applied to other domains. For open-domain event extraction, how to deal with noisy text like those social posts and how to organize temporal-spatial events are still great challenges in practice.

In what follows, we discuss possible future research directions for event extraction.

A. KNOWLEDGE-BOOSTED DEEP LEARNING

Although deep learning techniques have proven a powerful tool in automatic feature learning for event extraction, those neural models are normally with too many learning parameters as well as network configurations, which not only demand huge amount of annotated raw data but also require careful tuning of numerous network configurations. On the other hand, pattern matching can well exploit experts' knowledge for specifying accurate event patterns, though with the cost of increased human efforts. Although the two approaches seem contrary to each other, a promising direction can be that how to boost deep learning model with the inclusion of experts' knowledge. While an initial try can be envisioned as to use patterns in the neural model training process, more efforts need to be devoted to enjoying high-quality human knowledge.

B. DOMAIN-ADAPTIVE TRANSFER LEARNING

In closed-domain event extraction, various event schemas are usually predefined with detailed event types and event arguments' roles. Although schemas provide structured representation for events, clearly defining schemas involves domain-specific knowledge. Furthermore, domain-specific schemas are not easily extended from one domain to other domains, although many arguments with different labels share similar roles even in different types of events. Transfer learning seems to be a promising approach for developing domain-adaptive event extraction systems, where an extraction model trained in one domain can be easily applied to other domain with only a few of adjustments. Nonetheless, it could be an insightful attempt to first transfer the commonly trained models for multi-type classification into multi-task learning models.

C. RESOURCE-AWARE EVENT CLUSTERING

In open-domain event extraction, events are firstly detected from different sources, and event mentions or keywords are next extracted. Then event clustering is often performed according to the similarities in between event keywords, that is, a kind of topic-centered event clustering is enforced. However, we notice that not only events can be briefly described by its keywords, but also other resources in texts like publisher, On the one hand, we agree that topic-centered event clustering should still be the basic operation for open-domain event extraction; On the other hand, we note that with the inclusion of other resources, multi-focus clustering can be implemented, which might further prompt other tasks, like event reasoning for answering why such events happening, and event inference for answering what kind of next events being expected.

REFERENCES

[1] *Ace (Automatic Content Extraction) English Annotation Guidelines for Events*, Linguistic Data Consortium, Philadelphia, PA, USA, 2005.
[2] M. Rospocher, M. van Erp, P. Vossen, A. Fokkens, I. Aldabe, G. Rigau, A. Soroa, T. Ploeger, and T. Bogaard, "Building event-centric knowledge graphs from news," *J. Web Semantics*, vols. 37–38, pp. 132–151, Mar. 2016.
[3] Z. Li, X. Ding, and T. Liu, "Constructing narrative event evolutionary graph for script event prediction," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 4201–4207.
[4] S. J. Conlon, A. S. Abrahams, and L. L. Simmons, "Terrorism information extraction from online reports," *J. Comput. Inf. Syst.*, vol. 55, no. 3, pp. 20–28, 2015.
[5] M. Atkinson, J. Piskorski, H. Tanev, E. V. der Goot, R. Yangarber, and V. Zavarella, "Automated event extraction in the domain of border security," in *Proc. Int. Conf. User Centric Media*, 2009, pp. 321–326.
[6] H. Tanev, J. Piskorski, and M. Atkinson, "Real-time news event extraction for global crisis monitoring," in *Proc. Int. Conf. Appl. Natural Lang. Inf. Syst.*, 2008, pp. 207–218.
[7] M. Atkinson, M. Du, J. Piskorski, H. Tanev, R. Yangarber, and V. Zavarella, "Techniques for multilingual security-related event extraction from online news," in *Computational Linguistics*. Berlin, Germany: Springer, 2013, pp. 163–186.
[8] J. Piskorski, H. Tanev, and P. O. Wennerberg, "Extracting violent events from on-line news for ontology population," in *Proc. Int. Conf. Bus. Inf. Syst.*, 2007, pp. 287–300.
[9] W. Nuij, V. Milea, F. Hogenboom, F. Frasincar, and U. Kaymak, "An automated framework for incorporating news into stock trading strategies," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 4, pp. 823–835, Apr. 2014.
[10] P. Capet, T. Delavallade, T. Nakamura, A. Sandor, C. Tarsitano, and S. Voyatzi, "A risk assessment system with automatic extraction of event types," in *Proc. Int. Conf. Intell. Inf. Process.*, 2008, pp. 220–229.
[11] A. S. Abrahams, "Developing and executing electronic commerce applications with occurences," Ph.D. dissertation, Dept. Comput. Sci. Technol., Univ. Cambridge, Cambridge, U.K. Sep. 2002.
[12] J. A. Vanegas, S. Matos, F. González, and J. L. Oliveira, "An overview of biomolecular event extraction from scientific documents," *Comput. Math. Methods Med.*, vol. 2015, Aug. 2015, Art. no. 571381.
[13] B. M. Sundheim and N. A. Chinchor, "Survey of the message understanding conferences," in *Proc. Workshop Hum. Lang. Technol.*, 1993, pp. 56–60.
[14] L. Hirschman, "The evolution of evaluation: Lessons from the message understanding conferences," *Comput. Speech & Lang.*, vol. 12, no. 4, pp. 281–305, 1998.
[15] J. Allan, *Topic Detection and Tracking: Event-Based Information Organization*, vol. 12. Berlin, Germany: Springer, 2012.
[16] H. Yu, Y. Zhang, L. Ting, and L. Sheng, "Topic detection and tracking review," *J. Chin. Inf. Process.*, vol. 6, no. 21, pp. 77–79, 2007.
[17] J. Aguilar, C. Beller, P. McNamee, B. V. Durme, S. Strassel, Z. Song, and J. Ellis, "A comparison of the events and relations across ace, ere, tac-kbp, and framenet annotation standards," in *Proc. 2nd Workshop EVENTS Definition Detection Conf. Represent.*, 2014, pp. 45–53.
[18] Z. Song, A. Bies, S. Strassel, T. Riese, J. Mott, J. Ellis, J. Wright, S. Kulick, N. Ryant, and X. Ma, "From light to rich ere: Annotation of entities, relations, and events," in *Proc. 3rd Workshop EVENTS Definition, Detection Conf. Represent.*, 2015, pp. 89–98.
[19] T. Mitamura, Z. Liu, and E. Hovy, "Overview of TAC KBP 2015 event nugget track," in *Proc. Text Anal. Conf.*, 2015, pp. 1–11.
[20] C. Nédellec, R. Bossy, J.-D. Kim, J.-J. Kim, T. Ohta, S. Pyysalo, and P. Zweigenbaum, "Overview of BioNLP shared task 2013," in *Proc. BioNLP Shared Task Workshop*, 2013, pp. 1–7.
[21] J. Pustejovsky, P. Hanks, R. Sauri, A. See, R. Gaizauskas, A. Setzer, D. Radev, B. Sundheim, D. Day, L. Ferro, and M. Lazo, "The timebank corpus," in *Corpus linguistics*. Lancaster, U.K.: Lancaster Univ., 2003, p. 40.
[22] F. Hogenboom, F. Frasincar, U. Kaymak, and F. D. Jong, "An overview of event extraction from text," in *Proc. Workshop Detection Represent. Exploitation Events Semantic Web (DeRiVE)*, 2011, pp. 48–57.
[23] F. Hogenboom, F. Frasincar, U. Kaymak, F. D. Jong, and E. Caron, "A Survey of event extraction methods from text for decision support systems," *Decis. Support Syst.*, vol. 85, pp. 12–22, May 2016.
[24] Z. Zhang, Y. Wu, and Z. Wang, "A survey of open domain event extraction," Dept. Comput. Sci., Univ. Illinois Urbana-Champaign, Champaign, IL, USA, Tech. Rep. 01, 2018.

[25] M. Mejri and J. Akaichi, "A survey of textual event extraction from social networks," in *Proc. 1st Conf. Lang. Process. Knowl. Manage.*, 2017, pp. 1–13.
[26] F. Atefeh and W. Khreich, "A survey of techniques for event detection in twitter," *Comput. Intell.*, vol. 31, no. 1, pp. 132–164, 2015.
[27] D. Ahn, "The stages of event extraction," in *Proc. Workshop Annotating Reasoning About Time Events*, 2006, pp. 1–8.
[28] F. Petroni, N. Raman, T. Nugent, A. Nourbakhsh, Z. Panić, S. Shah, and J. L. Leidner, "An extensible event extraction system with cross-media event resolution," in *Proc. 24th SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 626–635.
[29] H. Yang, Y. Chen, K. Liu, Y. Xiao, and J. Zhao, "Dcfee: A document-level chinese financial event extraction system based on automatically labeled training data," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics Syst. Demonstrations*, 2018, pp. 50–55.
[30] S. Han, X. Hao, and H. Huang, "An event-extraction approach for business analysis from online Chinese news," *Electron. Commerce Res. Appl.*, vol. 28, pp. 244–260, Mar./Apr. 2018.
[31] J. Piskorski, H. Tanev, M. Atkinson, E. V. D. Goot, and V. Zavarella, "Online news event extraction for global crisis surveillance," in *Transactions on Computational Collective Intelligence*. vol. 6910. Berlin, Germany: Springer, 2011, pp. 182–212.
[32] S. Ribeiro, O. Ferret, and X. Tannier, "Unsupervised event clustering and aggregation from newswire and Web articles," in *Proc. EMNLP Workshop Natural Lang. Process. Meets Journalism*, 2017, pp. 62–67.
[33] S. Yu and B. Wu, "Exploiting structured news information to improve event detection via dual-level clustering," in *Proc. IEEE 3rd Int. Conf. Data Sci. Cyberspace (DSC)*, Jun. 2018, pp. 873–880.
[34] M. Liu, Y. Liu, L. Xiang, X. Chen, and Q. Yang, "Extracting key entities and significant events from online daily news," in *Proc. Int. Conf. Intell. Data Eng. Automated Learn.*, 2008, pp. 201–209.
[35] M. Naughton, N. Kushmerick, and J. Carthy, "Event extraction from heterogeneous news sources," in *Proc. AAAI Workshop Event Extraction Synth.*, 2006, pp. 1–6.
[36] W. Wang, Y. Ning, H. Rangwala, and N. Ramakrishnan, "A multiple instance learning framework for identifying key sentences and detecting events," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, 2016, pp. 509–518.
[37] A. Chouigui, O. B. Khiroun, and B. Elayeb, "A TF-IDF and co-occurrence based approach for events extraction from arabic news corpus," in *Proc. Int. Conf. Appl. Natural Lang. Inf. Syst.*, 2018, pp. 272–280.
[38] A. Ritter, Mausam, O. Etzioni, and S. Clark, "Open domain event extraction from twitter," in *Proc. 18th SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 1104–1112, doi: http://www.cse.iitd.ac.in/ mausam/index.html.
[39] D. Zhou, X. Zhang, and Y. He, "Event extraction from Twitter using non-parametric Bayesian mixture model with word embeddings," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics Long Papers*, 2017, pp. 808–817.
[40] D. Zhou, L. Chen, and Y. He, "A simple Bayesian modelling approach to event extraction from Twitter," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 700–705.
[41] D. Zhou, L. Chen, and Y. He, "An unsupervised framework of exploring events on Twitter: Filtering, extraction and categorization," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2468–2474.
[42] J. Weng and B.-S. Lee, "Event detection in twitter," in *Proc. 5th Int. AAAI Conf. Weblogs Social Media*, 2011, pp. 401–408.
[43] A. Guille and C. Favre, "Mention-anomaly-based event detection and tracking in Twitter," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*, Aug. 2014, pp. 375–382.
[44] N. Chambers and D. Jurafsky, "Template-based information extraction without the templates," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics Hum. Lang. Technol.*, 2011, pp. 976–986.
[45] L. Huang, T. Cassidy, X. Feng, H. Ji, C. R. Voss, J. Han, and A. Sil, "Liberal event extraction and event schema induction," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 258–268.
[46] Q. Yuan, X. Ren, W. He, C. Zhang, X. Geng, L. Huang, H. Ji, C.-Y. Lin, and J. Han, "Open-schema event profiling for massive news corpora," in *Proc. 27th Int. Conf. Inf. Knowl. Manage.*, 2018, pp. 587–596.
[47] D. Rusu, J. Hodson, and A. Kimball, "Unsupervised techniques for extracting and clustering complex events in news," in *Proc. 2nd Workshop EVENTS Definition Detection Conf. Represent.*, 2014, pp. 26–34.
[48] H.-W. Chun, Y.-S. Hwang, and H.-C. Rim, "Unsupervised event extraction from biomedical literature using co-occurrence information and basic patterns," in *Proc. 1st Int. Conf. Natural Lang. Process.*, 2004, pp. 777–786.
[49] A. Romadhony, D. H. Widyantoro, and A. Purwarianti, "Utilizing structured knowledge bases in open IE based event template extraction," *Appl. Intell.*, vol. 49, no. 1, pp. 206–219, 2019.
[50] H. Ji, "Cross-lingual predicate cluster acquisition to improve bilingual event extraction by inductive learning," in *Proc. Workshop Unsupervised Minimally Supervised Learn. Lexical Semantics*, 2009, pp. 27–35.
[51] X. Liu, H. Huang, and Y. Zhang, "Open domain event extraction using neural latent variable models," 2019, *arXiv:1906.06947*. [Online]. Available: https://arxiv.org/abs/1906.06947
[52] G. Glavaš and J. Šnajder, "Recognizing identical events with graph Kernels," in *Proc. 51st Annu. Meeting Assoc. Comput. Linguistics*, 2013, pp. 797–803.
[53] J. Pustejovsky, B. Ingria, R. Sauri, J. Castano, J. Littman, R. Gaizauskas, A. Setzer, G. Katz, and I. Mani, "The specification language TimeML," in *The Language Of Time: A Reader*. Oxford, U.K.: Oxford Univ. Press, 2005, pp. 545–557.
[54] J. Pustejovsky, J. M. Castano, R. Ingria, R. Sauri, R. J. Gaizauskas, A. Setzer, G. Katz, and D. R. Radev, "TimeML: Robust specification of event and temporal expressions in text," *Directions Question Answering*, vol. 3, pp. 28–34, Jan. 2003.
[55] *Rich Ere Annotation Guidelines Overview*, Linguistic Data Consortium, Philadelphia, PA, USA, 2016.
[56] J. Allan, J. G. Carbonell, G. Doddington, J. Yamron, and Y. Yang, "Topic detection and tracking pilot study final report," in *Proc. Broadcast News Transcription Understand. Workshop (DARPA)*, 1998, pp. 194–218.
[57] H. Meng, "Research on event extraction technology in the field of unexpected events," M.S. thesis, School Comput. Eng. Sci., Shanghai Univ., Shanghai, China, Jan. 2015.
[58] X. Ding, F. Song, B. Qin, and T. Liu, "Research on typical event extraction method in the field of music," *J. Chin. Inf. Process.*, vol. 25, no. 2, pp. 15–21, 2011.
[59] E. Riloff, "Automatically constructing a dictionary for information extraction tasks," in *Proc. 11th Nat. Conf. Artif. Intell.*, 1993, pp. 811–816.
[60] W. Lehnert, "Symbolic/subsymbolic sentence analysis: Exploiting the best of two worlds," *Advances in Connectionist and Neural Computation Theory*, vol. 1. Bristol, U.K.: Intellect, May 1994, pp. 135–164.
[61] K. B. Cohen, K. Verspoor, H. L. Johnson, C. Roeder, P. V. Ogren, W. A. Baumgartner, Jr., E. White, H. Tipney, and L. Hunter, "High-precision biological event extraction with a concept recognizer," in *Proc. Workshop Current Trends Biomed. Natural Lang. Process. Shared Task*, 2009, pp. 50–58.
[62] A. Casillas, A. D. D. Ilarraza, K. Gojenola, M. Oronoz, and G. Rigau, "Using Kybots for extracting events in biomedical texts," in *Proc. BioNLP Shared Task Workshop*, 2011, pp. 138–142.
[63] Q.-C. Bui and P. M. A. Sloot, "A robust approach to extract biomedical events from literature," *Bioinformatics*, vol. 28, no. 20, pp. 2654–2661, 2012.
[64] Q.-C. Bui, D. Campos, E. M. van Mulligen, and J. Kors, "A fast rule-based approach for biomedical event extraction," in *Proc. BioNLP Shared Task Workshop*, 2013, pp. 104–108.
[65] J. Borsje, F. Hogenboom, and F. Frasincar, "Semi-automatic financial events discovery based on lexico-semantic patterns," *Int. J. Web Eng. Technol.*, vol. 6, no. 2, pp. 115–140, 2010.
[66] E. Arendarenko and T. Kakkonen, "Ontology-based information and event extraction for business intelligence," in *Proc. Int. Conf. Artif. Intell. Methodol. Syst. Appl.*, 2012, pp. 89–102.
[67] L. Hunter, Z. Lu, J. Firby, W. A. Baumgartner, Jr., H. L. Johnson, P. V. Ogren, and K. B. Cohen, "OpenDMAP: An open source, ontology-driven concept analysis engine, with applications to capturing knowledge regarding protein transport, protein interactions and cell-type-specific gene expression," *BMC Bioinf.*, vol. 9, no. 1, p. 78, 2008.
[68] K. Cao, X. Li, W. Ma, and R. Grishman, "Including new patterns to improve event extraction systems," in *Proc. 31st Int. Flairs Conf.*, 2018, pp. 487–492.
[69] M.-V. Tran, M.-H. Nguyen, S.-Q. Nguyen, M.-T. Nguyen, and X.-H. Phan, "VnLoc: A real—Time news event extraction framework for vietnamese," in *Proc. 4th Int. Conf. Knowl. Syst. Eng.*, Aug. 2012, pp. 161–166.

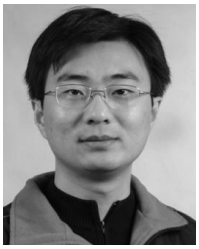
[70] A. Saroj, R. K. Munodtiya, and S. Pal, "Rule based event extraction system from newswires and social media text in Indian languages (eventxtract-il) for English and hindi data," in *Proc. FIRE Work. Notes*, 2018, pp. 302–307.
[71] M. A. Valenzuela-Escárcega, G. Hahn-Powell, M. Surdeanu, and T. Hicks, "A domain-independent rule-based framework for event extraction," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process.*, 2015, pp. 127–132.
[72] J.-T. Kim and D. I. Moldovan, "Acquisition of linguistic patterns for knowledge-based information extraction," *IEEE Trans. Knowl. Data Eng.*, vol. 7, no. 5, pp. 713–724, Oct. 1995.
[73] C. Aone and M. Ramos-Santacruz, "REES: A large-scale relation and event extraction system," in *Proc. 6th Conf. Appl. Natural Lang. Process.*, 2000, pp. 76–83.
[74] E. Riloff and J. Shoen, "Automatically acquiring conceptual patterns without an annotated corpus," in *Proc. 3rd Workshop Very Large Corpora*, 1995, pp. 148–161.
[75] R. Yangarber, R. Grishman, P. Tapanainen, and S. Huttunen, "Automatic acquisition of domain knowledge for information extraction," in *Proc. 18th Conf. Comput. Linguistics*, 2000, pp. 940–946.
[76] F. Li, H. Sheng, and D. Zhang, "Event pattern discovery from the stock market bulletin," in *Proc. Int. Conf. Discovery Sci.*, 2002, pp. 310–315.
[77] J. Jiang, "An event IE pattern acquisition method," *Comput. Eng.*, vol. 31, no. 15, pp. 96–98, 2005.
[78] S. Liao and R. Grishman, "Filtered ranking for bootstrapping in event extraction," in *Proc. COLING*, 2010, pp. 680–688.
[79] K. Cao, X. Li, M. Fan, and R. Grishman, "Improving event detection with active learning," in *Proc. Int. Conf. Recent Adv. Natural Lang. Process.*, 2015, pp. 72–77.
[80] P. Li, G. Zhou, and Q. Zhu, "Minimally supervised chinese event extraction from multiple views," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 16, no. 2, p. 13, 2016.
[81] F. Xu, H. Uszkoreit, and H. Li, "Automatic event and relation detection with seeds of varying complexity," in *Proc. AAAI Workshop Event Extraction Synth.*, 2006, pp. 12–17.
[82] A. Vlachos, P. Buttery, D. Ó. Séaghdha, and T. Briscoe, "Biomedical event extraction without training data," in *Proc. Workshop Current Trends Biomed. Natural Lang. Process. Shared Task*, 2009, pp. 37–40.
[83] H. Kilicoglu and S. Bergler, "Syntactic dependency based heuristics for biological event extraction," in *Proc. Workshop Current Trends Biomed. Natural Lang. Process. Shared Task*, 2009, pp. 119–127.
[84] H. Kilicoglu and S. Bergler, "Effective bio-event extraction using trigger words and syntactic dependencies," *Comput. Intell.*, vol. 27, no. 4, pp. 583–609, 2011.
[85] H. L. Chieu and H. T. Ng, "A maximum entropy approach to information extraction from semi-structured and free text," in *Proc. 18th Nat. Conf. Artif. Intell.*, 2002, pp. 786–791.
[86] J. Björne, J. Heimonen, F. Ginter, A. Airola, T. Pahikkala, and T. Salakoski, "Extracting complex biological events with rich graph-based feature sets," in *Proc. Workshop Current Trends Biomed. Natural Lang. Process. Shared Task*, 2009, pp. 10–18.
[87] M. Miwa, R. Sætre, J.-D. Kim, and J. Tsujii, "Event extraction with complex event classification using rich features," *J. Bioinf. Comput. Biol.*, vol. 8, no. 1, pp. 131–146, 2010.
[88] M. Miwa, P. Thompson, and S. Ananiadou, "Boosting automatic event extraction from the literature using domain adaptation and coreference resolution," *Bioinformatics*, vol. 28, no. 13, pp. 1759–1765, 2012.
[89] J. Björne, F. Ginter, S. Pyysalo, J. Tsujii, and T. Salakoski, "Complex event extraction at PubMed scale," *Bioinformatics*, vol. 26, no. 12, pp. i382–i390, 2010.
[90] K. Hakala, S. V. Landeghem, T. Salakoski, Y. V. de Peer, and F. Ginter, "EVEX in ST'13: Application of a large-scale text mining resource to event extraction and network construction," in *Proc. BioNLP Shared Task Workshop*, 2013, pp. 26–34.
[91] J. Xia, A. C. Fang, and X. Zhang, "A novel feature selection strategy for enhanced biomedical event extraction using the Turku system," *BioMed Res. Int.*, vol. 2014, Apr. 2014, Art. no. 205239.
[92] D. Martinez and T. Baldwin, "Word sense disambiguation for event trigger word detection in biomedicine," *BMC Bioinf.*, vol. 12, no. 2, p. S4, Mar. 2011.
[93] A. Majumder, A. Ekbal, and S. K. Naskar, "Biomolecular event extraction using a stacked generalization based classifier," in *Proc. 13th Int. Conf. Natural Lang. Process.*, 2016, pp. 55–64.
[94] R. Grishman, D. Westbrook, and A. Meyers, "NYU's English ACE 2005 system description," in *Proc. ACE Eval. Workshop*, vol. 5, 2005, PP. 1–7.
[95] X. Q. Pham, M. Q. Le, and B. Q. Ho, "A hybrid approach for biomedical event extraction," in *Proc. BioNLP Shared Task Workshop*, 2013, pp. 121–124.
[96] L. Sha, J. Liu, C.-Y. Lin, S. Li, B. Chang, and Z. Sui, "Rbpb: Regularization-based pattern balancing method for event extraction," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 1224–1234.
[97] A. Meyers, M. Kosaka, S. Sekine, R. Grishman, and S. Zhao, "Parsing and glarfing," in *Proc. RANLP*, 2001, pp. 1–7.
[98] Y.-Y. Zhao, B. Qin, and W.-X. Che, and T. Liu, "Research on chinese event extraction," *J. Chin. Inf. Process.*, vol. 22, no. 1, pp. 3–8, 2008.
[99] Z. Chen and H. Ji, "Language specific issue and feature exploration in Chinese event extraction," in *Proc. Hum. Lang. Technol. Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2009, pp. 209–212.
[100] P. Li, G. Zhou, Q. Zhu, and L. Hou, "Employing compositional semantics and discourse consistency in Chinese event extraction," in *Proc. Joint Conf. Empirical Methods Natural Lang. Process. Comput. Natural Lang. Learn.*, 2012, pp. 1006–1016.
[101] Y. Kitagawa, M. Komachi, E. Aramaki, N. Okazaki, and H. Ishikawa, "Disease event detection based on deep modality analysis," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process. Student Res. Workshop*, 2015, pp. 28–34.
[102] H. Ji and R. Grishman, "Refining event extraction through cross-document inference," in *Proc. 46th Annu. Meeting Assoc. Comput. Linguistics Hum. Lang. Technol.*, 2008, pp. 254–262.
[103] S. Liao and R. Grishman, "Using document level cross-event inference to improve event extraction," in *Proc. 48th Annu. Meeting Assoc. Comput. Linguistics*, 2010, pp. 789–797.
[104] P. Li, Q. Zhu, and G. Zhou, "Argument inference from relevant event mentions in Chinese argument extraction," in *Proc. 51st Annu. Meeting Assoc. Comput. Linguistics*, 2013, pp. 1477–1487.
[105] S. Liu, K. Liu, S. He, and J. Zhao, "A probabilistic soft logic based approach to exploiting latent and global information in event classification," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 2993–2999.
[106] Y. Hong, J. Zhang, B. Ma, J. Yao, G. Zhou, and Q. Zhu, "Using cross-entity inference to improve event extraction," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics Hum. Lang. Technol.*, 2011, pp. 1127–1136.
[107] S. Liao and R. Grishman, "Acquiring topic features to improve event extraction: In pre-selected and balanced collections," in *Proc. Int. Conf. Recent Adv. Natural Lang. Process.*, 2011, pp. 9–16.
[108] R. Huang and E. Riloff, "Modeling textual cohesion for event extraction," in *Proc. 26th AAAI Conf. Artif. Intell.*, 2012, pp. 1664–1670.
[109] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.
[110] P. Li, Q. Zhu, H. Diao, and G. Zhou, "Joint modeling of trigger identification and event type determination in Chinese event extraction," in *Proc. COLING*, 2012, pp. 1635–1652.
[111] P. Li, Q. Zhu, and G. Zhou, "Joint modeling of argument identification and role determination in chinese event extraction with discourse-level information," in *Proc. 23rd Int. Joint Conf. Artif. Intell.*, 2013, pp. 2120–2126.
[112] C. Chen and V. Ng, "Joint modeling for chinese event extraction with rich linguistic features," in *Proc. COLING*, 2012, pp. 529–544.
[113] Q. Li, H. Ji, and L. Huang, "Joint event extraction via structured prediction with global features," in *Proc. 51st Annu. Meeting Assoc. Comput. Linguistics*, 2013, pp. 73–82.
[114] A. Judea and M. Strube, "Event extraction as frame-semantic parsing," in *Proc. 4th Joint Conf. Lexical Comput. Semantics*, 2015, pp. 159–164.
[115] M. Collins, "Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2002, pp. 1–8.
[116] D. Das, N. Schneider, D. Chen, and N. A. Smith, "Probabilistic frame-semantic parsing," in *Proc. Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2010, pp. 948–956.
[117] S. Riedel, H.-W. Chun, T. Takagi, and J. Tsujii, "A Markov logic approach to bio-molecular event extraction," in *Proc. Workshop Current Trends Biomed. Natural Lang. Process. Shared Task*, 2009, pp. 41–49.
[118] S. Riedel, R. Sætre, H.-W. Chun, T. Takagi, and J. Tsujii, "Bio-molecular event extraction with Markov logic," *Comput. Intell.*, vol. 27, no. 4, pp. 558–582, 2011.

[119] D. Venugopal, C. Chen, V. Gogate, and V. Ng, "Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 831–843.
[120] S. Riedel and A. McCallum, "Fast and robust joint models for biomedical event extraction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2011, pp. 1–12.
[121] K. Yoshikawa, S. Riedel, T. Hirao, M. Asahara, and Y. Matsumoto, "Coreference based event-argument relation extraction on biomedical text," *J. Biomed. Semantics*, vol. 2, no. 5, p. S6, 2011.
[122] A. Vlachos and M. Craven, "Biomedical event extraction from abstracts and full papers using search-based structured prediction," in *Proc. BioNLP Shared Task Workshop*, 2011, pp. 36–40.
[123] D. Zhou and Y. He, "Biomedical events extraction using the hidden vector state model," *Artif. Intell. Med.*, vol. 53, no. 3, pp. 205–213, 2011.
[124] D. McClosky, M. Surdeanu, and C. D. Manning, "Event extraction as dependency parsing," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics Hum. Lang. Technol.*, 2011, pp. 1626–1635.
[125] X. Liu, A. Bordes, and Y. Grandvalet, "Biomedical event extraction by multi-class classification of pairs of text entities," in *Proc. BioNLP Shared Task Workshop*, 2013, pp. 45–49.
[126] L. Li, S. Liu, M. Qin, Y. Wang, and D. Huang, "Extracting biomedical event with dual decomposition integrating word embeddings," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 13, no. 4, pp. 669–677, Jul./Aug. 2016.
[127] S. Riedel and A. McCallum, "Robust biomedical event extraction with dual decomposition and minimal domain adaptation," in *Proc. BioNLP Shared Task Workshop*, 2011, pp. 46–50.
[128] Q. Li, H. Ji, H. Yu, and S. Li, "Constructing information networks using one single model," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1846–1851.
[129] A. Judea and M. Strube, "Incremental global event extraction," in *Proc. 26th Int. Conf. Comput. Linguistics Tech. Papers*, 2016, pp. 2279–2289.
[130] B. Yang and T. M. Mitchell, "Joint extraction of events and entities within a document context," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Hum. Lang. Technol.*, 2016, pp. 289–299.
[131] J. Araki and T. Mitamura, "Joint event trigger identification and event coreference resolution with structured perceptron," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 2074–2080.
[132] W. Kongburan, P. Padungweang, W. Krathu, and J. H. Chan, "Enhancing metabolic event extraction performance with multitask learning concept," *J. Biomed. Inform.*, vol. 93, May 2019, Art. no. 103156.
[133] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," 2016, *arXiv:1603.01360*. [Online]. Available: https://arxiv.org/abs/1603.01360
[134] W.-T. Yih, X. He, and C. Meek, "Semantic parsing for single-relation question answering," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 643–648.
[135] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil, "Learning semantic representations using convolutional neural networks for Web search," in *Proc. 23rd Int. Conf. World Wide Web*, 2014, pp. 373–374
[136] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1746–1751.
[137] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," 2014, *arXiv:1404.2188*. [Online]. Available: https://arxiv.org/abs/1404.2188
[138] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, Aug. 2011.
[139] D. Zeng, K. Liu, S. Lai, G. Zhou, and J. Zhao, "Relation classification via convolutional deep neural network," in *Proc. COLING*, 2014, pp. 2335–2344.
[140] T. H. Nguyen and R. Grishman, "Relation extraction: Perspective from convolutional neural networks," in *Proc. 1st Workshop Vector Space Modeling Natural Lang. Process.*, 2015, pp. 39–48.
[141] T. H. Nguyen and R. Grishman, "Event detection and domain adaptation with convolutional neural networks," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process.*, 2015, pp. 365–371.
[142] S. Liu, Y. Chen, K. Liu, J. Zhao, Z. Luo, and W. Luo, "Improving event detection via information sharing among related event types," in *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*. Cham, Switzerland: Springer, 2017, pp. 122–134.
[143] W. Lu and T. H. Nguyen, "Similar but not the same: Word sense disambiguation improves event detection via neural representation matching," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 4822–4828.
[144] S. Liu, R. Cheng, X. Yu, and X. Cheng, "Exploiting contextual information via dynamic memory network for event detection," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 1030–1035.
[145] H. Lin, Y. Lu, X. Han, and L. Sun, "Nugget proposal networks for chinese event detection," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, 2018, pp. 1565–1574.
[146] H. Lin, Y. Lu, X. Han, and L. Sun, "Cost-sensitive regularization for label confusion-aware event detection," 2019, *arXiv:1906.06003*. [Online]. Available: https://arxiv.org/abs/1906.06003
[147] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: https://arxiv.org/abs/1301.3781
[148] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.
[149] Y. Chen, L. Xu, K. Liu, D. Zeng, and J. Zhao, "Event extraction via dynamic multi-pooling convolutional neural networks," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process.*, 2015, pp. 167–176.
[150] T. H. Nguyen and R. Grishman, "Modeling skip-grams for event detection with convolutional neural networks," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 886–891.
[151] Z. Zhang, W. Xu, and Q. Chen, "Joint event extraction based on skip-window convolutional neural networks," in *Natural Language Understanding and Intelligent Applications*. Cham, Switzerland: Springer, 2016, pp. 324–334.
[152] G. Burel, H. Saif, M. Fernandez, and H. Alani, "On semantics and deep learning for event detection in crisis situations," in *Proc. Workshop Semantic Deep Learn. (SemDeep)*, 2017, pp. 1–12.
[153] L. Li, Y. Liu, and M. Qin, "Extracting biomedical events with parallel multi-pooling convolutional neural networks," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, to be published.
[154] D. Kodelja, R. Besançon, and O. Ferret, "Exploiting a more global context for event detection through bootstrapping," in *Proc. Eur. Conf. Inf. Retr.*, 2019, pp. 763–770.
[155] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. 11th Annu. Conf. Int. Speech Commun. Assoc.*, 2010, pp. 1045–1048.
[156] N. Limsopatham and N. Collier, "Bidirectional LSTM for named entity recognition in Twitter messages," in *Proc. 2nd Workshop Noisy User-Generated Text (WNUT)*, 2016, pp. 145–152.
[157] P. Wang, Y. Qian, F. K. Soong, L. He, and H. Zhao, "Part-of-speech tagging with bidirectional long short-term memory recurrent neural network," 2015, *arXiv:1510.06168*. [Online]. Available: https://arxiv.org/abs/1510.06168
[158] M. Miwa and M. Bansal, "End-to-end relation extraction using lstms on sequences and tree structures," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 1105–1116.
[159] J. Cross and L. Huang, "Incremental parsing with minimal features using bi-directional LSTM," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 32–37.
[160] X. Ma and E. Hovy, "End-to-end sequence labeling via Bi-directional LSTM-CNNs-CRF," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 1064–1074.
[161] T. H. Nguyen, K. Cho, and R. Grishman, "Joint event extraction via recurrent neural networks," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Hum. Lang. Technol.*, 2016, pp. 300–309.
[162] R. Ghaeini, X. Fern, L. Huang, and P. Tadepalli, "Event nugget detection with forward-backward recurrent neural networks," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 369–373.
[163] Y. Chen, S. Liu, S. He, K. Liu, and J. Zhao, "Event extraction via bidirectional long short-term memory tensor neural networks," in *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*. Cham, Switzerland: Springer, 2016, pp. 190–203.
[164] S. Yan and K.-C. Wong, "Context awareness and embedding for biomedical event extraction," 2019, *arXiv:1905.00982*. [Online]. Available: https://arxiv.org/abs/1905.00982
[165] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1724–1734.

[166] L. Sha, F. Qian, B. Chang, and Z. Sui, "Jointly extracting event triggers and arguments by dependency-bridge RNN and tensor-based argument interaction," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 5916–5923.
[167] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," in *Proc. ACL*, 2015, pp. 1556–1566.
[168] W. Zhang, X. Ding, and T. Liu, "Learning target-dependent sentence representations for chinese event detection," in *Proc. China Conf. Inf. Retr.*, 2018, pp. 251–262.
[169] D. Li, L. Huang, H. Ji, and J. Han, "Biomedical event extraction based on knowledge-driven tree-LSTM," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Hum. Lang. Technol.*, 2019, pp. 1421–1430.
[170] S. Duan, R. He, and W. Zhao, "Exploiting document level information to improve event detection via recurrent neural networks," in *Proc. 8th Int. Joint Conf. Natural Lang. Process.*, 2017, pp. 352–361.
[171] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. ICML*, 2014, pp. 1188–1196.
[172] T. M. Nguyen and T. H. Nguyen, "One for all: Neural joint modeling of entities and events," in *Proc. 33rd AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 6851–6858.
[173] Y. Zhang, G. Xu, Y. Wang, X. Liang, L. Wang, and T. Huang, "Empower event detection with bi-directional neural language model," *Knowl.-Based Syst.*, vol. 167, pp. 87–97, Mar. 2019.
[174] T. Lei, Y. Zhang, S. I. Wang, H. Dai, and Y. Artzi, "Simple recurrent units for highly parallelizable recurrence," 2017, *arXiv:1709.02755*. [Online]. Available: https://arxiv.org/abs/1709.02755
[175] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.
[176] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," 2019, *arXiv:1901.00596*. [Online]. Available: https://arxiv.org/abs/1901.00596
[177] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," 2019, *arXiv:1812.08434*. [Online]. Available: https://arxiv.org/abs/1812.08434
[178] S. Rao, D. Marcu, K. Knight, and H. D. III, "Biomedical event extraction using abstract meaning representation," in *Proc. BioNLP*, 2017, pp. 126–135.
[179] X. Liu, Z. Luo, and H. Huang, "Jointly multiple events extraction via attention-based graph information aggregation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 1247–1256.
[180] T. H. Nguyen and R. Grishman, "Graph convolutional networks with argument-aware pooling for event detection," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 5900–5907.
[181] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider, "Abstract meaning representaiton for sembanking," in *Proc. ACL*, 2013, pp. 178–186.
[182] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, 2017, pp. 1–14.
[183] Y. Zeng, B. Luo, Y. Feng, and D. Zhao, "WIP event detection system at TAC KBP 2016 event nugget track," in *Proc. Text Anal. Conf.*, 2016, pp. 1–5.
[184] Y. Zeng, H. Yang, Y. Feng, Z. Wang, and D. Zhao, "A convolution BiLSTM neural network model for chinese event extraction," in *Natural Language Understanding and Intelligent Applications*, vol. 10102. Cham, Switzerland: Springer, 2016, p. 275.
[185] X. Feng, L. Huang, D. Tang, H. ji, B. Qin, and T. Liu, "A language-independent neural network for event detection," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 66–71.
[186] W. Liu, Z. Yang, and Z. Liu, "Chinese event recognition via ensemble model," in *Proc. Int. Conf. Neural Inf. Process.*, 2018, pp. 255–264.
[187] A. Kuila, S. C. Bussa, and S. Sarkar, "A neural network based event extraction system for indian languages," in *Proc. FIRE (Working Notes)*, 2018, pp. 302–307.
[188] A. Nurdin and N. Maulidevi, "5W1H information extraction with CNN-bidirectional LSTM," *J. Phys. Conf. Ser.*, vol. 978, no. 1, p. 12078, 2018.
[189] Y. Liu, Q. Li, X. Liu, and L. Si, "Document information assisted event trigger detection," in *Proc. IEEE Int. Conf. Big Data*, Dec. 2018, pp. 5383–5385.
[190] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversatiral networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
[191] Y. Hong, W. Zhou, J. Zhang, G. Zhou, and Q. Zhu, "Self-regulation: Employing a generative adversarial network to improve event detection," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, 2018, pp. 515–526.
[192] J. Liu, Y. Chen, and K. Liu, "Exploiting the ground-truth: An adversarial imitation based knowledge distillation approach for event detection," in *Proc. 33rd AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 6754–6761.
[193] T. Zhang, H. Ji, and A. Sil, "Joint entity and event extraction with generative adversarial imitation learning," *Data Intell.*, vol. 1, no. 2, pp. 99–120, 2019.
[194] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to alight and translate," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–15.
[195] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[196] D. Hu, "An introductory survey on attention mechanisms in NLP problems," 2018, *arXiv:1811.05544*. [Online]. Available: https://arxiv.org/abs/1811.05544
[197] S. Liu, Y. Chen, K. Liu, and J. Zhao, "Exploiting argument information to improve event detection via supervised attention mechanisms," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, 2017, pp. 1789–1798.
[198] W. Wu, X. Zhu, J. Tao, and P. Li, "Event detection via recurrent neural network and argument prediction," in *Proc. CCF Int. Conf. Natural Lang. Process. Chin. Comput.*, 2018, pp. 235–245.
[199] W. Orr, P. Tadepalli, and X. Fern, "Event detection with neural networks: A rigorous empirical evaluation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 999–1004.
[200] J. Zhang, W. Zhou, Y. Hong, J. Yao, and M. Zhang, "Using entity relation to improve event detection via attention mechanism," in *Proc. CCF Int. Conf. Natural Lang. Process. Chin. Comput.*, 2018, pp. 171–183.
[201] R. Ding and Z. Li, "Event extraction with deep contextualized word representation and multi-attention layer," in *Proc. Int. Conf. Adv. Data Mining Appl.*, 2018, pp. 189–201.
[202] Y. Wu and J. Zhang, "Chinese event extraction based on attention and semantic features: A bidirectional circular neural network," *Future Internet*, vol. 10, no. 10, p. 95, Sep. 2018.
[203] Y. Zhao, X. Jin, Y. Wang, and X. Cheng, "Document embedding enhanced event detection with hierarchical and supervised attention," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, 2018, pp. 414–419.
[204] Y. Chen, H. Yang, K. Liu, J. Zhao, and Y. Jia, "Collective event detection via a hierarchical and bias tagging networks with gated multi-level attention mechanisms," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 1267–1276.
[205] S. Mehta, M. R. Islam, H. Rangwala, and N. Ramakrishnan, "Event detection using hierarchical multi-aspect attention," in *Proc. World Wide Web Conf.*, 2019, pp. 3079–3085
[206] J. Liu, Y. Chen, K. Liu, and J. Zhao, "Event detection via gated multilingual attention mechanism," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 4865–4872.
[207] Y. Li, C. Li, W. Xu, and J. Li, "Prior knowledge integrated with self-attention for event detection," in *Proc. China Conf. Inf. Retr.*, 2018, pp. 263–273.
[208] S. Abney, "Bootstrapping," in *Proc. 40th Annu. Meeting Assoc. Comput. Linguistics*, 2002, pp. 360–367.
[209] S. Liao and R. Grishman, "Can document selection help semi-supervised learning?: A case study on event extraction," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics Hum. Lang. Technol.*, 2011, pp. 260–265.
[210] J. Ferguson, C. Lockard, D. Weld, and H. Hajishirzi, "Semi-supervised event extraction with paraphrase clusters," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Hum. Lang. Technol.*, 2018, pp. 359–364.
[211] X. Wang, X. Han, Z. Liu, M. Sun, and P. Li, "Adversarial training for weakly supervised event detection," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Hum. Lang. Technol.*, 2019, pp. 998–1008.
[212] S. Liao and R. Grishman, "Using prediction from sentential scope to build a pseudo co-testing learner for event extraction," in *Proc. 5th Int. Joint Conf. Natural Lang. Process.*, 2011, pp. 714–722.
[213] T. Strohman, D. Metzler, H. Turtle, and W. B. Croft, "Indri: A language model-based search engine for complex queries," in *Proc. Int. Conf. Intell. Anal.*, 2005, vol. 2, no. 6, pp. 2–6.
[214] J. Wang, Q. Xu, H. Lin, Z. Yang, and Y. Li, "Semi-supervised method for biomedical event extraction," *Proteome Sci.*, vol. 11, no. 1, p. S17, 2013.

[215] M. Miwa, S. Pyysalo, T. Ohta, and S. Ananiadou, "Wide coverage biomedical event extraction using multiple partially overlapping corpora," *BMC Bioinf.*, vol. 14, no. 1, p. 175, 2013.
[216] T. H. Nguyen, L. Fu, K. Cho, and R. Grishman, "A two-stage approach for extending event detection to new types via neural networks," in *Proc. 1st Workshop Represent. Learn. NLP*, 2016, pp. 158–165.
[217] H. Peng, Y. Song, and D. Roth, "Event detection and co-reference with minimal supervision," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 392–402.
[218] L. Huang, H. Ji, K. Cho, I. Dagan, S. Riedel, and C. Voss, "Zero-shot transfer learning for event extraction," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, 2018, pp. 2160–2170.
[219] W. Li, D. Cheng, L. He, Y. Wang, and X. Jin, "Joint event extraction based on hierarchical event schemas from framenet," *IEEE Access*, vol. 7, pp. 25001–25015, 2019.
[220] S. Liu, Y. Chen, S. He, K. Liu, and J. Zhao, "Leveraging framenet to improve automatic event detection," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 2134–2143.
[221] Y. Chen, S. Liu, X. Zhang, K. Liu, and J. Zhao, "Automatically labeled data generation for large scale event extraction," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, 2017, pp. 409–419.
[222] A. Kimmig, S. Bach, M. Broecheler, B. Huang, and L. Getoor, "A short introduction to probabilistic soft logic," in *Proc. NIPS Workshop Probabilistic Program. Found. Appl.*, 2012, pp. 1–4.
[223] Y. Zeng, Y. Feng, R. Ma, Z. Wang, R. Yan, C. Shi, and D. Zhao, "Scale up event extraction learning via automatic training data generation," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 6045–6052.
[224] J. Araki and T. Mitamura, "Open-domain event detection using distant supervision," in *Proc. COLING*, 2018, pp. 878–891.
[225] S. Zheng, W. Cao, W. Xu, and J. Bian, "Doc2EDAG: An end-to-end document-level framework for chinese financial event extraction," 2019, *arXiv:1904.07535*. [Online]. Available: https://arxiv.org/abs/1904.07535
[226] K. Reschke, M. Jankowiak, M. Surdeanu, C. Manning, and D. Jurafsky, "Event extraction using distant supervision," in *Proc. 9th Int. Conf. Lang. Resour. Eval.*, 2014, pp. 4527–4531.
[227] Z. Zhu, S. Li, G. Zhou, and R. Xia, "Bilingual event extraction: A case study on trigger type determination," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 842–847.
[228] Z. Chen and H. Ji, "Can one language bootstrap the other: A case study on event extraction," in *Proc. NAACL HLT Workshop Semi-Supervised Learn. Natural Lang. Process.*, 2009, pp. 66–74.
[229] A. Hsi, J. G. Carbonell, and Y. Yang, "Modeling event extraction via multilingual data sources," in *Proc. Text Anal. Conf.*, 2015, pp. 1–8.
[230] A. Hsi, Y. Yang, J. Carbonell, and R. Xu, "Leveraging multilingual training for limited resource event extraction," in *Proc. COLING*, 2016, pp. 1201–1210.
[231] M. Li, Y. Lin, J. Hoover, S. Whitehead, C. Voss, M. Dehghani, and H. Ji, "Multilingual entity, relation, event and human value extraction," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2019, pp. 110–115.
[232] Y. Yang, T. Pierce, and J. G. Carbonell, "A study on retrospective and on-line event detection," in *Proc. 21st Annu. Int. SIGIR Conf. Res. Develop. Inf. Retr.*, 1982, pp. 28–36.
[233] R. Nallapati, A. Feng, F. Peng, and J. Allan, "Event threading within news topics," in *Proc. 13th Int. Conf. Inf. Knowl. Manage.*, 2004, pp. 446–453.
[234] N. Stokes and J. Carthy, "Combining semantic and syntactic document classifiers to improve first story detection," in *Proc. 24th Annu. Int. SIGIR Conf. Res. Develop. Inf. Retr.*, 2001, pp. 424–425.
[235] Z. Li, B. Wang, M. Li, and W.-Y. Ma, "A probabilistic model for retrospective news event detection," in *Proc. 28th Annu. Int. SIGIR Conf. Res. Develop. Inf. Retr.*, 2005, pp. 106–113.
[236] T. Ge, L. Cui, B. Chang, Z. Sui, and M. Zhou, "Event detection with burst information networks," in *Proc. COLING*, 2016, pp. 3276–3286.
[237] J. Morris and G. Hirst, "Lexical cohesion computed by thesaural relations as an indicator of the structure of text," *Comput. Linguistics*, vol. 17, no. 1, pp. 21–48, Mar. 1991.
[238] Z. Lei, L.-D. Wu, Y. Zhang, and Y. C. Liu, "A system for detecting and tracking Internet news event," in *Proc. Pacific-Rim Conf. Multimedia*, 2005, pp. 754–764.
[239] C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*. Cambridge, MA, USA: MIT Press, 1999.
[240] B. Jans, S. Bethard, I. Vulić, and M. F. Moens, "Skip n-grams and ranking functions for predicting script events," in *Proc. 13th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2012, pp. 336–344.
[241] F. M. Zanzotto, M. Pennacchiotti, and K. Tsioutsiouliklis, "Linguistic redundancy in Twitter," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2011, pp. 659–669.
[242] S. Petrovic, M. Osborne, R. McCreadie, C. Macdonald, I. Ounis, and L. Shrimpton, "Can twitter replace newswire for breaking news?" in *Proc. 7th Int. Conf. Weblogs Social Media*, 2013.
[243] I. Mani and G. Wilson, "Robust temporal processing of news," in *Proc. 38th Annu. Meeting Assoc. Comput. Linguistics*, 2000, pp. 69–76.
[244] A. X. Chang and C. Manning, "Sutime: A library for recognizing and normalizing time expressions," in *Proc. 8th Int. Conf. Lang. Resour. Eval.*, 2012, pp. 3735–3740.

WEI XIANG received the B.S. and M.Eng. degrees from the School of Electronic Information and Communications, Huazhong University of Science and Technology (HUST), Wuhan, China, in 2011 and 2015, respectively. He is currently working as a Lecturer with the School of Electronic Information and Communications, HUST, with research focuses on natural language processing and knowledge graph.



BANG WANG received the B.S. and M.S. degrees from the Department of Electronics and Information Engineering, Huazhong University of Science and Technology (HUST), Wuhan, China, in 1996 and 2000, respectively, and the Ph.D. degree from the Electrical and Computer Engineering (ECE) Department, National University of Singapore (NUS), in 2004. He is currently working as a Professor with the School of Electronic Information and Communications, HUST. His research interests include machine intelligence, network science, and social computing technologies.